M. Sc. Engineering Thesis

# Algorithms for Constructing Multi-labeled Phylogenetic Trees from Quartets with Fewer Leaf Duplications 

By<br>Md. Mizanur Rahman<br>Student no.: 0413052008

Submitted to
Department of Computer Science and Engineering in partial fulfilment of the requirements for the degree of Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering Bangladesh University of Engineering and Technology (BUET) Dhaka-1000, Bangladesh

The thesis titled "Algorithms for Constructing Multi-labeled Phylogenetic Trees from Quartets with Fewer Leaf Duplications", submitted by Md. Mizanur Rahman, Roll No. 0413052008, Session April 2013, to the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Engineering and approved as to ts style and contents. Examination held on 12th November, 2017.

## Board of Examiners

1. $\qquad$

Dr. Md. Saidur Rahman
Professor
Chairman
Professor (Supervisor)
Department of CSE
BUET, Dhaka 1000.
2. Nhlren

Dr. M. Sohel Rahman
Member
Professor \& Head
(Ex-officio)
Department of CSE
BUET, Dhaka 1000.
3.


Dr. Md. Abul Kashem Mia
Member
Professor
Department of CSE
BUET, Dhaka 1000.
4.


Dr. Md. Shamsuzzoha Bayzid
Member
Assistant Professor
Department of CSE
BUET, Dhaka 1000.


Dr. Saifuddin Md. Tareeq
Member
Professor
(External)
Department of CSE
University of Dhaka, Dhaka-1000.

## Candidate's Declaration

This is to certify that the work presented in this thesis entitled "Algorithms for Constructing Multi-labeled Phylogenetic Trees from Quartets with Fewer Leaf Duplicatrons" is the outcome of the investigation carried out by me under the supervision of Professor Dr. Md. Saidur Rahman in the Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology (BUET), Dhaka. It is also declared that neither this thesis nor any part thereof has been submitted or is being currently submitted anywhere else for the award of any degree or diploma.
Md. Mizanur Rahman

Candidate

## Contents

Board of Examiners ..... ii
Candidate's Declaration ..... iii
Acknowledgements ..... ix
Abstract ..... x
1 Introduction ..... 1
1.1 Problem Definition ..... 2
1.2 Applications of Phylogenies ..... 3
1.3 Applications of Multi-labeled Phylogenetic Trees ..... 5
1.3.1 Perusing Host-Parasite Cospeciation ..... 5
1.3.2 Constructing Gene Trees ..... 6
1.3.3 Biogeography ..... 6
1.3.4 Constructing Phylogenetic Networks ..... 7
1.4 Motivation ..... 8
1.5 Literature Review ..... 9
1.6 Objectives of this Thesis ..... 13
1.7 Summary of Results ..... 13
1.8 Thesis Organization ..... 14
1.9 Summary ..... 15
CONTENTS ..... v
2 Preliminaries ..... 16
2.1 Basic Terminology ..... 16
2.1.1 Trees ..... 16
2.1.2 Phylogenetic Trees ..... 17
2.1.3 Supertrees ..... 20
2.1.4 Caterpillar ..... 21
2.1.5 Multi-labeled Phylogenetic Trees ..... 21
2.1.6 Gene Trees and Species Trees ..... 22
2.1.7 Phylogenetic Networks ..... 23
2.1.8 Triplets and Quartets ..... 24
2.1.9 Splits and Super-splits ..... 25
2.1.10 Depth one Chains ..... 26
2.2 Bipartition ..... 27
2.2.1 Method of Bipartition ..... 27
2.3 Summary ..... 32
3 Algorithms for Constructing MUL-Trees ..... 33
3.1 Algorithms ..... 33
3.1.1 RDCRD Approach ..... 34
3.1.2 $\mathrm{RDCS}_{\text {Split }} \mathrm{A}$ Approach ..... 34
3.1.3 BDCRD Approach ..... 37
3.1.4 QMUL Approach ..... 39
3.1.5 Advanced QMUL(AQMUL) Approach ..... 42
3.2 Super-split Analysis Technique ..... 46
3.2.1 Consistency Checking Method ..... 46
3.2.2 Super-split Method ..... 47
3.3 Quartet Analysis Technique ..... 48
3.3.1 Consistency Checking Method ..... 49
3.3.2 Quartet Matching Method ..... 49
3.4 Summary ..... 50
4 Performance of Algorithms ..... 51
4.1 Experiments on Simulated Datasets ..... 52
4.1.1 Simulated Datasets ..... 52
4.1.2 Simulated Data Results ..... 53
4.1.3 Computational Issues ..... 57
4.2 Experiments on Real Datasets ..... 58
4.2.1 Real Datasets ..... 58
4.2.2 Real Data Results ..... 60
4.3 Duplication Vs Consistency ..... 63
4.4 Summary ..... 64
5 Conclusion ..... 65
A Supporting Information ..... 68
List of Publications ..... 69
Index ..... 74

## List of Figures

1.1 Illustration of (a) set of quartets and (b) MUL-tree. ..... 3
1.2 Illustration of (a) a quartet set $Q$ and (b) supertree $T$. ..... 8
1.3 Illustration of (a) a quartet set $Q$ and (b) supertree $T$. ..... 9
2.1 Illustration of a tree (a) unrooted (b) rooted. ..... 17
2.2 Illustration of phylogenetic tree (a) unrooted (b) rooted. ..... 18
2.3 Illustration of bifurcating and multifurcating tree (a) unrooted (b) rooted. ..... 19
2.4 Illustration of (a) quartet set $Q$ (b) supertree $T$. ..... 20
2.5 Illustration of caterpillar (a) unrooted (b) rooted. ..... 21
2.6 Illustration of Multi-labeled Phylogenetic tree (a) unrooted (b) rooted. ..... 22
2.7 Illustration of Phylogenetic Network (a) unrooted (b) rooted. ..... 23
2.8 Illustration of (a) Triplet and (b) Quartet. ..... 24
2.9 Illustration of quartet consistency with a tree $T$. ..... 25
2.10 Illustration of Super-split. ..... 26
2.11 Illustration of (a) Depth one element (b) Depth one chain. ..... 27
2.12 An example iteration of the Bipartition Algorithm. ..... 30
3.1 Illustration of RDCRD approach ..... 35
3.2 Illustration of $\operatorname{RDCS}_{\text {Split }} \mathrm{A}$ approach ..... 36
3.3 Illustration of BDCRD approach ..... 38
3.4 Illustration of QMUL approach ..... 41
3.5 Illustration of AQMUL approach ..... 43
3.6 Consistency of (a) quartets with respect to (b) depth one chain. ..... 46
3.7 Illustration of consistency checking method. ..... 47
3.8 Illustration of super-split method. ..... 48
3.9 Illustration of quartet matching method. ..... 50
4.1 A chart showing the difference in performance among five techniques in terms of number of duplications when $\mathrm{c}=90 \%$ and $\mathrm{q}=n^{1} .25$ ..... 54
4.2 Comparison of QMUL and AQMUL in terms of number of duplications under various model conditions. ..... 56
4.3 An original MUL-tree on flowering plants with 7 duplications. ..... 59
4.4 The obtained MUL-tree by applying AQMUL on the quartets extracted from the MUL-tree shown in Figure 4.3. This MUL-tree has 7 duplications. ..... 59
4.5 An original MUL-tree on violet species with 20 duplications. ..... 61
4.6 The obtained MUL-tree by applying AQMUL on the quartets extracted from the MUL-tree shown in Figure 4.5. This MUL-tree has 20 duplications. ..... 62
4.7 A chart showing the difference in performance among MTRT, QMUL and AQ- MUL on real datasets in terms of number of duplications. ..... 63

## Acknowledgements

First of all, I would like to declare that all the appraisals belong to the Almighty ALLAH.
I would like to express my deep gratitude to my supervisor Professor Dr. Md. Saidur Rahman for introducing me to the fascinating and prospective field of bioinformatics and phylogenetic trees. I have learned from him how to carry on a research work, how to write, speak and present well. I thank him for his patience in reviewing my so many inferior drafts, for correcting my proofs and language, suggesting new ways of thinking, leading to the right way and encouraging me to continue my research work.

I would like to thank all the members of the examination board, Prof. Dr. M. Sohel Rahman, Prof. Dr. Md. Abul Kashem Mia, Dr. Md. Shamsuzzoha Bayzid and Prof. Dr. Saifuddin Md. Tareeq for their valuable comments. I am very grateful to the ICT Division, Ministry of Posts, Telecommunications and Information Technology, Government of the People's Republic of Bangladesh for funding me during the research period.

I would also like to thank Dr. Md. Iqbal Hossain, Mrs. Shaheena Sultana, Mrs. Nazmun Nessa Moon, Mr. Md. Manzurul Hasan, Mr. Mohammad Al Mahmud, Mr. Abu Reyan Ahmed, Mr. Sujoy Das, Mr. Malay Devnath and all the members of my research group for their valuable suggestions and continual encouragements. I convey my heartfelt reverence to my parents and other family members for giving their best support throughout my work to overcome the tedium of repetitive trials to new findings.

Finally, every honor and every victory on earth is due to Allah, descended from Him and must be ascribed to Him. He has endowed me with good health and with the capability to complete this work. I deeply express my sincere gratitude to the endless kindness of Allah.

## Abstract

A central problem in computational and evolutionary biology is the inference of phylogenetic tree from a set of triplets or quartets. Researchers try to construct phylogenetic tree by combining many triplets or quartets into a single phylogenetic tree and this problem is known as supertree construction problem. The problem with supertree construction is that the constructed supertree does not always ensure the consistency of the entire input quartets from where it is designed and hence some important evolutionary information is lost. A possible solution of this problem is to construct Multi-labeled Phylogenetic Trees (MUL-trees) instead of supertree where all quartets are consistent. Let $Q=\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{k}\right\}$ be a collection of quartets over a set of taxa $S$. A MUL-tree $T$ over a set of quartets $Q$ is a tree where more than one of its leaves can be labeled by a single taxon and each quartet $q \epsilon Q$ are consistent with respect to the tree $T$. The problem of constructing a MUL-tree from a set of quartets is much more complex than that of standard phylogenetic trees. To the best of our knowledge, there is no study to construct MUL-trees from quartets to date.

In this thesis, we have proposed two algorithms with three auxiliary techniques to construct MUL-trees from a set of quartets. We first have proposed a bipartition based divide and conquer approach (QMUL) with super-split analysis technique. But it cannot always guarantee the minimum number of duplications. To overcome the shortcoming of QMUL approach we finally proposed AQMUL (MUL-trees from Quartets with Advanced method) approach which is the modification of QMUL approach. We have conducted experiments on simulated datasets and real datasets to analyze the performance of our proposed algorithms. We have found that AQMUL is more efficient than QMUL to construct MUL-Trees in terms of average number of duplications.

## Chapter 1

## Introduction

After publication of Charles Darwin's book On the Origin of Species; By means of natural selection, the theory of evolution was widely accepted. Since then remarkable developments in evolutionary studies brought the scientists to the phylogenetics, a field that studies the biological or the morphological data of species to output a mathematical model such as a tree known as phylogenetic tree; representing the evolutionary interrelationship of species and the process of their evolution. Besides, phylogenetics is not only limited to the biology but may also arise anywhere that the concept of evolution appears. Several approaches have been introduced to infer evolutionary relationships. Recently, a vast research effort has been devoted to phylogenetic tree reconstruction from quartets, a field denoted as quartet-based reconstruction among many others.

Quartet based phylogenetic tree inference has been receiving comprehensive attention from the researchers of evolutionary biology and mathematics. Different approaches have been proposed and improved time to time for constructing supertree from quartets $[1,2,4]$. There are various methods to design supertrees from quartets such as maximum quartet consistency (MQC), quartets max cut (QMC), short quartet puzzling (SQP), quartet amalgamation (QA) and matrix representation with parsimony (MRP). However constructing a supertree in which all quartets are consistent is not always possible [5]. This raises the problem of finding a tree in which the input is a set of quartets and a supertree is sought that displays the maximum
number of quartet: maximum quartet consistency (MQC) [10]. The Maximum Quartet Consistency (MQC) problem is an NP hard optimization problem [17]. One the other hands deciding whether there exists a supertree satisfying all the quartets in an arbitrary given set is NPcomplete [11]. Consequently the MQC problem cannot always ensure the consistency of the entire input quartets from where it was designed. Thereby we are losing important evolutionary information. When no such tree exists because of conflicts in the branching information, researchers may try to construct Multi-labeled Phylogenetic Trees (in short MUL-trees) by allowing leaf duplication in which all quartets are consistent.

In the rest of this chapter, we provide the necessary background and objectives for this study on Multi-labeled Phylogenetic Trees. In Section 1.1 we define the problem. We discuss the applications of Multi-labeled Phylogenetic Trees in Section 1.3. We devote Section 1.4 for the motivation of our works, Section 1.5 for the literature review and we detail the objective of this thesis in Section 1.6. Finally, Section 1.7 is a summary of this work and Section 1.8 is the description of the organization of this thesis.

### 1.1 Problem Definition

Let $Q=\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{k}\right\}$ be a collection of quartets over a set of taxa $S$. We have to construct a multi-labeled phylogenetic tree (MUL-tree) which is consistent with each of the quartets in a given set $Q$ over the set of taxa $S$ having more than one leaf with the same label. This problem is known as MUL-trees construction problem. Figure 1.1(a) depicts set of quartet $Q$ over a taxa set $S$ and Figure 1.1(b) depicts an unrooted MUL-tree over a quartet set $Q=$ $\left\{q_{1}, q_{2}, q_{3}, q_{4}, q_{5}, q_{6}, q_{7}\right\}$.

Problem: MUL-trees construction.
Input: A collection of quartets $Q=\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{k}\right\}$ over a set of taxa $S$.
Output: A multi-labeled phylogenetic tree $T$ from quartets $Q$ on taxa set $S$ with fewer leaf duplications.

(a)

(a)

(b)

Figure 1.1: Illustration of (a) set of quartets and (b) MUL-tree.

### 1.2 Applications of Phylogenies

Phylogenies are important because it enriches our understanding of how genes, genomes, species (and molecular sequences more generally) evolve. Through phylogenies, we learn not only how the sequences came to be the way they are today, but also general principles that enable us to predict how they will change in the future. This is not only of fundamental importance but also extremely useful for numerous applications. In this section, we give a brief overview of phylogeny and its applications from [26].

Phylogenies are reconstructed on the basis of character data, where a "character" is any feature of an organism that can have different states. A typical biological example of a character is a nucleotide position in a DNA sequence, with the character state being the particular nucleotide $(A, G, C, T)$ in occupying that position. From a mathematical standpoint, a character is just a function that maps the set of taxa to its set of states. Molecular phylogenetics research is concerned not only with the evolutionary history of different organisms, but also with how the different characters evolve in the course of that history.

A second common use of phylogenies is to test biogeographic hypothesis. Biogeography is concerned with the geographical distribution of organisms, extant and extinct. For example, a researcher may be interested in whether a particular species have colonized a set of islands a single time or repeatedly. This can be assessed by determining whether all of the species on the island arose from a single most recent mainland common ancestor or whether they are multiple independent mainland species.

One can also use a phylogeny to attempt to infer the amino acid sequence of extinct proteins. These putative extinct proteins can then be synthesized or an artificial gene coding for them can be produced, and the functional characteristics of the proteins that are of interest can be tested. In a more practical vein, phylogenies can be used to track the evolution of diseases, which can, in turn, be used to design drugs and vaccines that are more likely to be effective against the currently dominant strains. The most prominent example of this use is the flu vaccine, which is altered from year to year as medical experts work to keep track of the influenza types most likely to dominate in a given flu season.

Phylogenies based on sequence data provide us with more accurate descriptions of patterns of relatedness than was available before the advent of molecular sequencing. Phylogenies now inform the Linnaean classification of new species. Molecular sequencing technologies and phylogenetic approaches can be used to learn more about a new pathogen outbreak. This includes finding out about which species the pathogen is related to and subsequently the likely source of transmission. This can lead to new recommendations for public health policy. Phylogenies can help to inform conservation policy when conservation biologists have to make tough decisions about which species they try to prevent from becoming extinct.

Finally, phylogenies have even been used in criminal cases in order to assess DNA evidence presented in court cases to inform situations, e.g. where someone has committed a crime, when food is contaminated, or where the father of a child is unknown, most famously, in a case where a doctor in Louisiana was accused of having deliberately infected his girlfriend with HIV. The phylogenetic evidence featured prominently in the trial and the doctor was ultimately convicted of attempted second degree murder.

With the advent of newer, faster sequencing technologies, it is now possible to take a sequencing machine out to the field and sequence species of interest in situ. Phylogenies are needed to add biological meaning to the data. In summary, phylogenies are useful in any endeavor where the historical and hierarchical structure of the evolution of species can be used to infer the history of the point of interest.

### 1.3 Applications of Multi-labeled Phylogenetic Trees

In the predominant research of evolutionary chronicle, MUL-trees arise from the modeling of biological processes where it is necessary to use particular leaf labels more than once. For example, a gene tree can contain several leaves labeled by the same species due to gene duplication events [8]. As another example, area cladograms (used in biogeography studies), where the names of geographical areas are used to label the leaves, may apply the same label to more than a single leaf [18]. MUL-trees can also be useful for perusing host-parasite cospeciation [19]. MUL-trees come off naturally in, for example, biogeography and gene evolution studies. The other applications of MUL-trees include molecular systematic, biomedical science and computer science. Recently, MUL-trees have been used to construct phylogenetic networks representing the evolutionary history of polyploid species [12]. Intriguingly, MUL-trees and related structures have applications in other areas, including data mining [22], multi-labeled data analysis [23], and the construction of directed acyclic word graphs [24]. Due to the application of multi-labeled phylogenetic tree evolutionary scientists provide a modest attention to construct a MUL-tree from rooted triplets and unrooted quartets.

### 1.3.1 Perusing Host-Parasite Cospeciation

Our first example is an application of the MUL-tree in host-parasite cospeciation. If the association between two species is very close, they may speciate in parallel, such that speciation events in the two taxa are coupled. This is called cospeciation. Host-parasite cospeciation has long been of strong interest to evolutionary biologists. Cospeciation occurs when interacting groups, such as hosts and parasites, speciate in tandem, generating congruent phylogenies. Cospeciation can be a neutral process in which parasites speciate merely because they are isolated on diverging host islands. Adaptive evolution may also play a role, but this has seldom been tested. One of the most exciting areas of cospeciation analysis concerns comparing the rate of evolution in host and parasites and the timing of host-parasite cospeciation. The presence of multiple lineages of parasites on the same host can give rise to the concept of MUL-trees in
host-parasite cospeciation. So MUL-trees are useful for perusing host-parasite cospeciation.

### 1.3.2 Constructing Gene Trees

We now provide another application of the MUL-tree is to show the evolutionary history of gene trees. A genetree is a model of how a gene evolves through duplication, loss, and nucleotide substitution. As a gene at a locus in the genome replicates and its copies are passed on to more than one offspring, branching points are generated in the gene tree. Gene trees are leaf-labeled trees inferred from molecular sequences. Gene trees are usually multi-labeled, i.e. a single species can label more than one leaf, since duplication events resulted in the presence of several copies of the genes in the species genomes.

A gene tree is an evolutionary tree built by analyzing a gene family, i.e. homologous molecular sequences appearing in the genome of different organisms. Gene trees can provide evidence for gene duplication events, as well as speciation events. Sequences from different homologs can be included in a gene tree; the subsequent analyses should cluster orthologs, thus demonstrating the evolutionary history of the orthologs. Gene trees are primarily used to estimate species trees, i.e. trees displaying the evolutionary relationships among studied species. Unfortunately, most gene trees can significantly differ from the species tree for methodological or biological reasons, such as long branch attraction, lateral gene transfers, deep gene coalescence and, principally, gene duplications and losses [28]. For this reason, species trees are usually estimated from a large number of gene trees.

### 1.3.3 Biogeography

Biogeography is the study of the past and present distribution of species. Closely related species tend to be found in the same geographic region, but the same environments in distant regions are usually occupied by very different species. This could be the result of a species being separated and then developing to their environments in new areas. Identifying common patterns among area cladograms that arise in historical biogeography is an important tool for
biogeographical inference.
As a first step, these methods construct a general area cladogram by replacing the taxon label of the leaf with the label of the area in which the taxon is found. Note that some taxa may occur in more than one area (called widespread taxa) and there may be many taxa endemic to one area (called redundant taxa). This in turn translates to many leaves with the same area label or leaves with more than one area label in the general area cladogram. Hence, the area cladograms constructed as above do not, and cannot, represent a history of the areas. Consequently, the direct inference methods further process the general area cladograms to produce a branching history of areas where each leaf is labeled with a unique area, called resolved area cladograms.

### 1.3.4 Constructing Phylogenetic Networks

We now provide another important application of the MUL-tree to construct phylogenetic networks representing the evolutionary history of polyploid species. It is now common place to use molecular data to reconstruct the evolutionary past of species. Often such a phylogenetic analysis proceeds as follows: Molecular markers (such as genes or plastids) are sequenced from organisms of interest and, using some model of evolution, the evolutionary past of each marker is then reconstructed using one of the many tree building methods [5, 13]. The resulting leaflabelled phylogenetic tree is then used as an approximation to the evolutionary relationships between the organisms under consideration. The situation becomes more complicated when trying to unravel the evolutionary past of organisms that are suspected to have undergone hybridization. This commonly results in polyploids, that is, organisms having several genomes. Basically speaking, this situation is thought to have arisen either via genome duplication or through hybridization between species resulting in organisms containing copies of all of their parent's genomes. Consequently, the molecular markers commonly used in phylogenetic analysis have the potential of supporting conflicting evolutionary relationships. The challenge is therefore to reconcile these conflicting relationships into an overall evolutionary picture. In the study of the evolution of polyploids, multi-labelled trees, or trees having some leaves with
the same label, can arise. Lott et al. [16] suggested that rather than inferring a phylogenetic network directly, it may be easier to first reconcile the input into a single MUL-tree and then apply an algorithm to output a network with the minimum number of non-tree nodes.

### 1.4 Motivation

Consider a situation where quartets are input and we have to design a supertree from this quartet by supertree method. A tree $T$ is said to be the supertree of a set of quartets $Q$ if all the quartets in $Q$ can be found from tree $T$ by deleting leaves and contracting edges. Note that supertree is free from leaf duplication. Different methods available in the literature for constructing supertree from quartets. Among these, the most prominent approaches are, Maximum Quartet Consistency, Quartets MaxCut, Short Quartet Puzzling, Quartet amalgamation, Matrix Representation with Parsimony (Most commonly used). Consider the example in Figure 1.2 in this example all quartets $Q$ are consistent with respect to the supertree $T$ because we can get all quartets from tree $T$ by deleting leaves and contracting edges. But this is not always possible to design a supertree where all quartets are consistent.


Figure 1.2: Illustration of (a) a quartet set $Q$ and (b) supertree $T$.

Let see another example in Figure 1.3 to understand the actual scenario. Here $Q$ is a set of quartets we have to design a supertree $T$ from $Q$ which ensures the consistency of each quartet in $Q$ but it is not possible because some quartets in $Q$ contain conflicting branch information. At these situation researchers uses different techniques to design supertrees which ensure the consistency of maximum number of quartets and it is known as Maximum Quartet Consistency
(MQC) problem. Thus we have designed the supertree in Figure 1.3 from the quartet set $Q$ which ensure the consistency of five quartets among seven quartets. Here quartets $q_{3}$ and $q_{7}$ are inconsistent because quartets $q_{3}$ and $q_{7}$ contain conflicting branch information. Let see the quartet $q_{3}$ in supertree $T$ here the path from taxa 1 to taxa 3 overlaps the path from taxa 2 to taxa 4. Similarly in case of quartet $q_{7}$ in super tree $T$ the path from taxa 1 to taxa 4 overlaps the path from taxa 2 to taxa 5 .


(a)

(b)

Figure 1.3: Illustration of (a) a quartet set $Q$ and (b) supertree $T$.

So we can say that, supertree cannot always ensure the consistency of the entire input quartets from where it was designed. Thereby we are loosing of important evolutionary information. On the other hand in 1992 Michael Steel [11] showed that deciding whether there exists a super tree satisfying all the quartets in an arbitrary given set, is NP-complete. So overcome limitations of supertree method we can construct MUL-trees. Note that MUL-tree is also a supertree when duplication is zero.

### 1.5 Literature Review

Phylogenetic tree reconstruction has been receiving extensive attention from the researchers of evolutionary biology and mathematics. Different approaches have been proposed and improved time to time for constructing phylogenetic tree from quartets and triplets. Researcher constructs supertree, phylogenetic network and Multi-labeled phylogenetic tree from quartets or triplets. The problems associated with quartets or triplets consistency have been extensively studied. Since their inception, quartets or triplets consistency have raised several interesting problems,
and most of these problems have remained unsolved. Among the other problems, MUL-trees construction from quartets is an important concern.

Supertree methods are a fundamental and practical way of inferring phylogenies. Generally speaking, these methods amalgamate a collection of source" trees (quartets or triplets) on overlapping subsets of taxa into a single parent tree that contains the taxa of all of the source trees. This parent tree is called a supertree. Supertree methods represent one of the major ways by which the Tree of Life can be estimated, although many supertree methods have been developed in the last few decades, none has been shown to produce more accurate trees than the popular Matrix Representation with Parsimony (MRP) method.

In 2006 Bryant et al. [33] proved that compatibility of unrooted phylogenetic trees is fixed parameter tractable (FPT) and they also shown that there exists a linear time, FPT algorithm for compatibility of unrooted trees. They present an $O(n f(k))$ algorithm, proving that compatibility of unrooted phylogenetic trees is fixed parameter tractable (FPT) with respect to the number $k$ of trees. In this research they used a set of quartet as unrooted tree and design a compatible supertree using these quartets.

In 2008, Snir et al. [7] proposed a new quartet-based method, short quartet puzzling (SQP). The experimental studies in [7] shows that SQP provides more accurate trees than QP, NJ and MP. It differs from the previous techniques in that it does not require all three topologies of the quartets on every 4 taxa. It is able to construct the output tree from a subset of all possible quartets as input. This is a two-phase technique: the first phase uses the randomized technique for selecting input quartets from all possible 4 -trees (estimated using ML), and the second phase uses Quartet MaxCut (QMC) [7, 8] technique for combining quartets into a single tree.

Snir et al. [14] use an algorithm to design Supertrees from a set of quartet which is known as Quartet MaxCut (QMC). In this paper Snir and Rao presented Quartets MaxCut (QMC), a heuristic for MQC that can be applied to arbitrary sets of quartet trees (i.e., ones that may not contain a tree on every quartet). Quartet MaxCut is a quartet-based phylogenetic reconstruction method. The method is based on a recursive divide and conquer algorithm that seeks to maximize the ratio between satisfied and violated quartets at each step. This
task is performed by a very fast semi definite programming (SDP) like heuristic for solving MaxCut in a graph induced by the quartets. Snir and Rao showed that by encoding the source trees as quartet trees, QMC could be used as a supertree method for arbitrary inputs. Their study evaluated this QMC-based supertree method for a number of biological supertree profiles; however, since the true supertree was not known, they could not evaluate the topological accuracy of the supertrees they constructed. Instead, they computed the average similarity of the QMC and MRP supertrees to the source trees, using two different similarity measures. This comparison showed that QMC had higher average similarity to the source trees under one criterion, and lower average similarity with respect to another; thus, Snir and Rao failed to establish that QMC produced better trees than MRP. QMC is operating in polynomial time and providing no guarantees with respect to its optimization problem, MQC.

In 2014 Reaz et al. [1] proposed an algorithm Quartet FM (QFM) to construct accurate phylogenetic tree from a set of quartets. In this paper, they showed that QFM is a heuristic which uses a bipartition technique inspired from the famous Fiduccia and Mattheyses (FM) algorithm for partitioning a hyper graph minimizing the cut size. Their approach cannot ensure the consistency of all the quartets. In this paper they demonstrated the superiority of QFM method over QMC (Quartet MaxCut), which is known to be the best quartet amalgamation method to date.

The problem of recovering network-like evolutionary histories has recently attracted a considerable amount of attention in the literature. In 2006 Huber et al. [12] proposed an algorithm which takes as input a multi labeled phylogenetic tree and outputs a phylogenetic network with certain desirable properties to understand the origins of certain polyploids. The overall complexity of the algorithm is $O(n \log n)$.

The problem of constructing an optimal rooted phylogenetic network from an arbitrary set of rooted triplets is an NP-hard problem. In 2014 Poormohammadi et al. [7] proposed a heuristic algorithm TripNet, which tries to construct a rooted phylogenetic network with the minimum number of reticulation nodes from an arbitrary set of rooted triplets. They showed that in all cases TripNet outputs an appropriate rooted phylogenetic network in an acceptable
time.
Multi-labeled phylogenetic tree construction problems are concerned to design a MUL-tree from a set of quartets or triplets and which is consistent with respect to the set of quartets or triplets. In 2008 Huber et al. [8] proposed an algorithm to find the complexity of deriving Multi labeled trees from bipartitions. In this paper they showed that it is NP-hard to decide whether a collection of bipartitions of a multiset can be represented by a multi-labeled tree. They also show that it is possible to generalize to multi-labeled trees a well-known condition that characterizes when a collection of bipartitions encodes a phylogenetic tree. Using this generalization, they obtain a fixed-parameter algorithm for the complexity of deriving Multilabeled trees from bipartitions in terms of a parameter associated to the given multiset.

Smallest Multi-labeled Phylogenetic Tree (SMRT) is neoteric method to construct multi labeled phylogenetic tree from rooted triplets developed by Sylvain Guillemot, Jesper Jansson and Wing-Kin Sung in 2011 [34]. SMRT is a heuristic method that takes a set of rooted triplets over a leaf set as input and output a MUL-tree which is consistent with rooted triplets. They prove that even the restricted case of determining if there exists a MUL-tree consistent with the input and having just one leaf duplication is NP-hard. They also provide an exact algorithm for the problem running in exponential time and space where $n$ is the number of taxa.

In 2014 Hassanzadeh et al. [6] proposed a new triplet based method (MTRT) to construct a multi-labeled phylogenetic tree from rooted triplets. The results of MTRT show that triplets alone cannot provide enough information to infer the true MUL-tree. So, it is in appropriate to infer a MUL-tree using triplet information alone and considering the minimum number of duplications. MTRT is a heuristic method that aims to solve the SMRT problem. The goal of the algorithm is to construct a minimal MUL-tree that is consistent with the input set of triplets and minimizes the number of its duplications.

The MUL-trees construction approaches mentioned above were developed by researchers to construct MUL-trees from a set of triplet instead of a set of quartet. But to the best of our knowledge, there is no study to construct MUL-trees from quartets. In this paper we present an algorithm named QMUL and AQMUL to construct MUL-trees from quartets with fewer
leaf duplications. The QMUL and the AQMUL are conventional divide and conquer algorithm with some modification.

### 1.6 Objectives of this Thesis

The objective of this thesis is to provide algorithms for constructing multi-labeled phylogenetic trees from quartets to overcome the shortcoming of supertree construction problem mentioned in Section 1.1. We will study the problem related to quartet based phylogenetic tree construction. We will conduct experiments on simulated data and real data to analyze the performance of our propose algorithms. In conventional divide and conquer based MUL-tree construction approach, we will use the bipartition method which will ensure the consistency of maximum number of quartets. In our research we will also provide super-split analysis technique and quartet analysis technique which be the most decisive part of our propose algorithms. The super-split analysis technique and the quartet analysis technique will be used to perform duplication with fewer leaf. The main objectives of this thesis are as follows:

1. To study the problems on construction of supertrees and multi-labeled phylogenetic trees from a set of quartets.
2. To design algorithms for constructing MUL-trees with fewer leaf duplications.
3. Performance comparison of the developed algorithms.
4. To conduct experiments on synthetic data and real data to analyze the performance of propose algorithms.

### 1.7 Summary of Results

In this thesis, we basically address the limitations of supertree construction problem and proposed appropriate algorithms to solve the problem. We have proposed two algorithms with
three auxiliary techniques to construct MUL-trees from a set of quartets. We first have proposed a bipartition based divide and conquer approach (QMUL) with super-split analysis technique. But it cannot always guarantee the minimum number of duplications. To overcome the shortcoming of QMUL approach we finally proposed AQMUL (MUL-trees from Quartets with Advanced method) approach which is the modification of QMUL approach. We have conducted experiments on simulated datasets and real datasets to analyze the performance of our proposed algorithms. We have found that AQMUL is more efficient than QMUL to construct MUL-Trees in terms of average number of duplications. Our proposed auxiliary techniques to construct MUL-trees are are as follows:

1. Randomized divide and conquer approach with randomized duplication technique (in short RDCRD).
2. Bipartition based divide and conquer approach with randomized duplication technique (in short BDCRD).
3. Randomized divide and conquer approach with super-split analysis technique (in short $\left.\operatorname{RDCS}_{\text {Split }} \mathrm{A}\right)$.

### 1.8 Thesis Organization

The rest of the thesis is organized as follows. In Chapter 2 we discuss the relevant ideas and necessary definitions from phylogenetic tree and multi-labeled phylogenetic tree to understand our research work and which was used throughout the paper. We have also discussed bipartition method, define split, super-split and Depth one chain in this chapter. Chapter 3 describes the meathod and material to solve our problem. In this chapter we have also describe supersplit analysis technique and quartet analysis technuque which is the most crucial part of our algorithms. Chapter 4 is devote for simulation and performance analysis. Finally, We conclude in Chapter 5 with some open problems and future research scopes related to this thesis.

### 1.9 Summary

In this chapter we have characterized multi-labeled phylogenetic tree construction from a set of quartet. We have provided some applications of multi-labeled phylogenetic tree like perusing host-parasite cospeciation, design gene trees, biogeography and construct phylogenetic networks. We have also presented motivation and the objective of this thesis. We have introduced the similar problems found in the literature. Finally we have provided the results of this thesis and its organization.

## Chapter 2

## Preliminaries

In this chapter we give necessary definitions and terminologies which will be used throughout the thesis. Most of the contents of this chapter are taken from the existing literature $[1,4,5,6$, $15,21]$ in order to study the basic methodologies of analyzing MUL-trees. Definitions that are not included in this chapter will be introduced as they are needed. We start, in Section 2.1 by giving some definitions of standard phylogeny-theoretical terms used in our thesis. In Section 2.2 we define different terms related to bipartition technique inspired by the famous Fiduccia and Mattheyses (FM) algorithm for bipartitioning a hyper graph minimizing the cut size [9].

### 2.1 Basic Terminology

In this section we give some definitions of standard phylogenetic-theoretical terms used throughout the remainder of this thesis. Interested readers are referred to see the detailed texts of the literature $[1,4,5,6,15,21]$.

### 2.1.1 Trees

Since this thesis is about phylogenetic trees, it is therefore appropriate to start by defining a tree. A tree $S=\{V, E\}$ is a connected graph with no cycles. Figure 2.1 is an example of a
tree. The vertices in a tree are usually called nodes. A rooted tree is a tree in which one of the nodes is distinguished from the others; otherwise the tree is unrooted. The distinguished node is called the root of the tree. The root of a tree is generally drawn at the top. In Figure 2.1 (b), the root is $u_{1}$. Every node $u$ other than the root is connected by an edge to some other node $p$ called the parent of $u$. We also call $u$ a child of $p$. We draw the parent of a node above that node. For example, in Figure 2.1(b), $u_{1}$ is the parent of $u_{2}, v_{3}$ and $u_{3}$, while $u_{2}$ is the parent of $v_{1}$ and $v_{2} ; v_{4}$ and $v_{4}$ are children of $u_{3}$. A leaf is a node of a tree that has no children. An internal node is a node that has one or more children. Thus every node of a tree is either a leaf or an internal node. In Figure 2.1(b), the leaves are $v_{1}, v_{2}, v_{3}, v_{4}$ and $v_{5}$, and the nodes $u_{2}$ and $u_{3}$ are internal nodes.


Figure 2.1: Illustration of a tree (a) unrooted (b) rooted.

### 2.1.2 Phylogenetic Trees

Phylogenetics is the study of evolutionary relationships among organisms or genes. A graphical representation of evolutionary relationships among various biological species or other entities based upon similarities and differences in their physical or genetic characteristics is called phylogenetic tree or evolutionary tree. We can also say that a phylogenetic tree is a hypothesis that depicts the evolutionary relationships among groups of organisms; in detailed in phylogenetic trees, branch points indicate new species diverged from a common ancestor. Species (or groups of species) and their most recent common ancestor form a clade within a phylogenetic tree.

Technically speaking, such a tree is a simple, connected graph with no cycles, and it is leaflabeled in case each of its leaves (i.e. vertices of degree 1) is labeled by precisely one element from some set. The set of labels corresponds to the set of species, populations or organisms under consideration. Trees can be classified as unrooted or rooted phylogenetic trees.


Figure 2.2: Illustration of phylogenetic tree (a) unrooted (b) rooted.

An unrooted phylogenetic tree or just unrooted tree is an acyclic connected graph having no internal vertices of degree two and every leaf having different label. In Figure 2.2(a) the leaves are vertices of degree one. On the other hand a rooted phylogenetic tree in Figure 2.2(b) is similar to an unrooted tree, except it has one internal vertex of degree two, which is called the root. The internal vertices of unrooted or rooted (except the root) trees can have degree three or greater. For example a binary phylogenetic tree, is a tree having all internal vertices of degree three. Again the only exception is the root, which has degree two. In a fully resolved binary phylogenetic tree with $n$ leaf nodes there are $n-1$ internal nodes. The leaves of the tree represent species. For example let $L(T)$ be the set of leaves for tree $T$. If T the set of trees, then we can say that $L(T)$ is the union of the leaf sets of the trees in $T$. In a rooted tree we say that a vertex $x$ is an ancestor of a vertex $y$, if the path from $y$ to the root passes through $x$. We can also say that $y$ is the descendant of $x$. The vertices adjacent to a vertex that are descendants of the vertex are called the children of the vertex, and the adjacent vertex that is an ancestor is called the parent of that vertex.

There is a clear distinction between rooted and unrooted settings. Under the rooted setting, trees contain a distinguished internal node denoted as the root and phylogenetic relationships are expressed by the least common ancestor relation. In the unrooted setting, there is no such


Figure 2.3: Illustration of bifurcating and multifurcating tree (a) unrooted (b) rooted.
node and relationships are represented by splits internal edges that split the taxa set into two parts. Both rooted and unrooted phylogenetic trees can be either bifurcating or multifurcating , and either labeled or unlabeled. A rooted bifurcating tree has exactly two descendants arising from each interior node (that is, it forms a binary tree), and an unrooted bifurcating tree takes the form of an unrooted binary tree, a free tree with exactly three neighbors at each internal node. In contrast, a rooted multifurcating (or polytomous) tree may have more than two children at some nodes and an unrooted multifurcating tree may have more than three neighbors at some nodes. Rooted and unrooted bifurcating and multifurcating trees are shown in figure 2.3. A labeled tree has specific values assigned to its leaves, while an unlabeled tree, sometimes called a tree shape, defines a topology only. The number of possible trees for a given number of leaf nodes depends on the specific type of tree, but there are always more multifurcating than bifurcating trees, more labeled than unlabeled trees and more rooted than unrooted trees. The last distinction is the most biologically relevant; it arises because there are many places on an unrooted tree to put the root.

### 2.1.3 Supertrees

It often happens that several phylogenies with different but overlapping taxa sets have to be combined within a single phylogeny, representing a summary of these source phylogenies. The resulting tree is called a supertree as it is built from trees and usually contains more species or taxa than each input tree. Supertrees are phylogenies assembled from smaller phylogenies that share some but not necessarily all species or taxa (leaf nodes) in common. Thus, supertrees can make novel statements about relationships of taxa that do not co-occur on any single input tree while still retaining hierarchical information from the input trees. As a method of combining existing phylogenetic information, supertrees potentially solve many of the problems associated with other methods (e.g., absence of homologous characters, incompatible data types, or nonoverlapping sets of taxa). In addition to helping synthesize hypotheses of relationships among larger sets of taxa, supertrees can suggest optimal strategies for taxon sampling (either for future supertree construction or for experimental design issues such as choice of outgroups), can reveal emerging patterns in the large knowledge base of phylogenies currently in the literature, and can provide useful tools for comparative biologists who frequently have information about variation across much broader sets of taxa than those found in any one tree.


Figure 2.4: Illustration of (a) quartet set $Q$ (b) supertree $T$.

Mathematically we can say that a tree $T$ is said to be the supertree of a quartet set $Q=$ $\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{k}\right\}$ if all the quartets in $Q$ can be found from tree $T$ by deleting leaves and contracting edges. Supertrees are themselves phylogenetic trees, but they are built by combining a set of smaller phylogenetic trees. Figure 2.4 shows a super tree which is consistent with the quartet set $Q=\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{5}\right\}$.

### 2.1.4 Caterpillar

In graph theory, a caterpillar or caterpillar tree is a special type of tree in which all the vertices are within distance one of a central path. An unrooted caterpillar tree in Figure 2.5(a) has one central path with leaves branching of it. In a rooted caterpillar tree in Figure 2.5(b) leaves knot to a single path from the root to the single leaf. A binary caterpillar tree is a rooted binary tree where every internal node has at least one child which is a leaf. Caterpillar trees have minimum number of pendant pair. So a caterpillar is a tree which metamorphoses into a path when its leaves are removed.

(b)

Figure 2.5: Illustration of caterpillar (a) unrooted (b) rooted.

### 2.1.5 Multi-labeled Phylogenetic Trees

A multi-labeled tree is a natural generalization of the standard phylogenetic tree model that allows the same leaf label to be used more than once in a single tree structure. Basically there are two types of MUL-trees i). unrooted MUL-tree and ii). rooted MUL-tree.

An unrooted multi-labeled phylogenetic tree is a tuple $T=\{T, M, \psi\}$ consisting of an unrooted tree $T$, called underlying tree, a set of labels $M$, and a surjective labeling function $\varphi: L(T) \longrightarrow M$ that maps each leaf of $T$ with a label in $M$. A MUL-tree in which each leaf has the same label (i.e., $|M|=1$ ) is called a uniform MUL-tree. Informally, a MUL-tree is simply an unrooted phylogeny in which multiple leaves can have the same label. For any label $(l) \in M, \varphi^{-} 1(l)$ is the set of all leaves labeled. If $\varphi$ is a bijection, the corresponding unrooted MUL-tree is just a (singly-labeled) unrooted tree.

Similarly, a rooted multi-labeled phylogenetic tree is a tuple $T=\{T, M, \psi\}$ consisting of an


Figure 2.6: Illustration of Multi-labeled Phylogenetic tree (a) unrooted (b) rooted.
rooted tree $T$, a set of labels $M$, and a subjective labeling function $\varphi: L(T) \longrightarrow M$ that maps each leaf of $T$ with a label in $M$. Note that the difference between unrooted and rooted MULtrees are in the underlying tree which is a rooted or unrooted tree for rooted and unrooted MUL-trees, respectively. Figure 2.6 illustrate an unrooted and a rooted MUL-tree.

Number of duplications: For any MUL-tree $M$, denote the set of all leaf labels that occur in $M$ by $L(M)$. For any leaf label $x \epsilon L(M)$, the number of duplications of $x$ is equal to the number of occurrences of $x$ in $M$ minus 1 . The number of leaf duplications in $M$, denoted by $d(M)$, is the total number of duplications of all leaf labels in $L(M)$. Define $m(M)$ as the number of leaves in $M$. Then, $d(M)=m(M)-L(M)$. For example consider the MUL-tree in Figure 2.6(a), here the total number leaves in the MUL-tree is $m(M)=9$ and set of all leaf labels that occur in MUL-tree is $L(M)=7$; so the number of duplication $d(M)=m(M)-L(M)=9-7=2$.

### 2.1.6 Gene Trees and Species Trees

A gene tree is an evolutionary tree built by analyzing a gene family, i.e., homologous molecular sequences appearing in the genome of different organisms. A species tree is a tree which displaying the evolutionary relationships among studied species. Gene trees are usually multilabeled, i.e., a single species can label more than one leaf, since duplication events almost always result in the presence of several copies of the genes in the species genomes. Most gene trees can significantly differ from the species tree for methodological or biological reasons, such as


Figure 2.7: Illustration of Phylogenetic Network (a) unrooted (b) rooted.
long branch attraction, lateral gene transfers, incomplete lineage sorting, gene duplications and losses [25]. For this reason, species trees are usually estimated from a large number of gene trees.

### 2.1.7 Phylogenetic Networks

A phylogenetic network is any graph used to represent evolutionary relationships (either abstractly or explicitly) between a set of taxa that labels some of its nodes (usually the leaves) Explicit networks represent evolutionary events, especially reticular events like horizontal gene transfer, where a gene is transferred between two unrelated organisms. Similar to trees, networks can also be divided into two groups, namely unrooted networks and rooted networks. Both are defined analogously to unrooted and rooted trees. Unrooted networks can be compared to an unrooted tree: there is no root and the edges can be spread to all sides.

An unrooted phylogenetic network $N$ on $X$ is any unrooted graph whose leaves are bijectively labeled by the taxa in $X$. Rooted networks on the other hand are comparable to rooted trees. Their branches emerge from one root and are built up to a tree-like network. A rooted phylogenetic network $N$ on $X$ is a rooted $D A G$ (direct acyclic graph) whose set of leaves is bijective labeled by the taxa in $X$. Any node of indegree $\geq 2$ is called reticulate node and all others are called tree nodes. Any edge leading to a reticulate node is called a reticulate edge and all others are called tree edges. Consequentially, unrooted and rooted networks are similar to unrooted and rooted trees. The relevant difference is that networks include the representa-


Figure 2.8: Illustration of (a) Triplet and (b) Quartet.
tion of evolutionary events whilst trees don't. Figure 2.7 illustrate an unrooted and a rooted phylogenetic network.

### 2.1.8 Triplets and Quartets

The smallest informative piece of rooted phylogenetic information is called triplet. A rooted triplet, or triplet for short, is a binary rooted tree on three distinct taxa. A triplet on three taxa $x, y$ and $w$ is denoted by $(w(x y))$ if the lowest common ancestor of $x$ and $y$ is a proper descendant of that of $x$ and $w$, or $y$ and $w$ as shown in Figure 2.8. The smallest informative piece of unrooted phylogenetic information is called quartet. It is an undirected phylogenetic tree having exactly four taxa or leaves with a single, central and internal edge. A quartet over the taxa $w, x, y, z$ denoted as $((w, x),(y, z))$ whenever there is an edge in the underlying tree separating the pair $w, x$ from the pair $y, z$ as shown in Figure 2.8. A bipartition of an unrooted tree $T$ is formed by taking any edge in $T$, and writing down the two sets of taxa that would be formed by deleting that edge. Let $T$ be a tree over the taxa set $S$. Now, if we take an internal edge $e$ of $T$ and delete $e$, then we get two subtrees, namely, $T_{a}$ and $T_{b}$. Let $S_{a}$ and $S_{b}$ be the sets of taxa of $T_{a}$ and $T_{b}$ respectively. We shall denote such bipartition by $\left(S_{a}, S_{b}\right)$. Thus an internal edge in $T$ corresponds to a bipartition of $S$.

Given a quartet $((w, x),(y, z))$, we say that a quartet tree on this set is unresolved if it is a star (four edges, each touching a leaf) and denote it by (wxyz). If the quartet tree has an internal edge separating two pairs of leaves we say that it is resolved. We will use quartet in lieu of quartet tree or resolved quartet when the sense is clear. A quartet $q=((w, x),(y, z))$ is satisfied with respect to a bipartition $\left(S_{a}, S_{b}\right)$ if taxa $w$ and $x$ reside in one part and taxa


Figure 2.9: Illustration of quartet consistency with a tree $T$.
$y$ and $z$ reside in the other. The quartet $q$ is said to be violated with respect to a bipartition $\left(S_{a}, S_{b}\right)$ when taxa $w$ and $y$ (or $w$ and $z$ ) reside in one part and taxa $x$ and $z$ (or $x$ and $y$ ) reside in the other part. The quartet $q$ is said to be deferred with respect to a bipartition $\left(S_{a}, S_{b}\right)$ if any three of its four taxa reside in one part and the fourth one resides in the other. On the other hand quartet $q$ is said to be isolated with respect to a bipartition $\left(S_{a}, S_{b}\right)$ if all the four taxa reside in any one part. A satisfied quartet is consistent with respect to a tree $T$ that is formed from the bipartition $\left(S_{a}, S_{b}\right)$. A quartet $((w, x),(y, z))$ is consistent with respect to a tree $T$ if in tree $T$, the path from $w$ to $x$ does not intersect the path from $y$ to $z$ or if there is an edge (or path in general) separating the taxa pair $(w, x)$ from $(y, z)$. In Figure 2.9 among the three quartets, quartet $q_{1}$ is consistent with a tree $T$ as there exists an edge in $T$ such that it separates $(w, x)$ from $(y, z)$. Other two quartet $q_{2}$ and $q_{3}$ are inconsistent with $T$ as no such edge exists in $T$. A taxon $t_{1}$ of a quarte $q_{1}$ is inconsistent if the location of $t_{1}$ makes the quartet inconsistent with respect to a tree $T$ and the taxon $t_{1}$ is known as inconsistent taxon. The taxa pair which contains inconsistent taxon is called inconsistent pair. For example in Figure 2.9(a) the taxon $x$ makes the quartet $q_{2}$ to inconsistent, so $x$ is inconsistent taxon and the taxa pair $(x, z)$ is inconsistent pair.

### 2.1.9 Splits and Super-splits

Let $T$ be an unrooted tree and $e$ be an edge of $T$. Removal of $e$ divides $T$ into two components. Let $A$ be the set of leaves in one component and $B$ be the set of leaves in another component. $A$ and $B$ are called the parts of the resulting split $A \mid B$. Order does not matter, so $A \mid B$ is
identical to $B \mid A$. Each edge in a tree induces a unique split. A split is called quartet if each of $A$ and $B$ contains exactly two leaves. For example a split $w, x \mid y, z$ or $((w, x),(y, y))$ is a quartet. A split $A \mid B$ is called a super-split $\left(S_{\text {split }}\right)$ of a quartet set $Q$ over a taxa set $S$ if $A$ and $B$ contain all the leaves in $S$ and each quartet can be derived by deleting leave and contracting edge from $A \mid B$ or if all the quartets in $Q$ displayed by $S_{\text {split }}$. Let consider a quartet set $Q=\{((1,3),(2,4)),((1,4),(2,5))\}$. Each quartet in set $Q$ can be derived from the splits $\{((1,3,4),(2,4,5))\}$ or $\{((1,2,3,5),(1,2,4))\}$ and all leaves in $S$ also present in splits; so $\{((1,3,4),(2,4,5))\}$ or $\{((1,2,3,5),(1,2,4))\}$ is a super-split of quartet set $Q$. A super-split has two parts; the left part and the right part. The part of a super-split which contains minimum number of taxa is called minimum super-element. In Figure 2.10 there are two super-splits and three minimum super-elements, $(1,3,4),(2,4,5)$ and $(1,2,4)$.


Figure 2.10: Illustration of Super-split.

### 2.1.10 Depth one Chains

A tree $T$ over taxa set $S$ is said to be a star star, if $T$ has only one internal node and there is an edge from the internal node incident to each taxon. We shall refer to such a tree as a depth one tree. The set of taxa which return depth one tree is refer to as depth one element in short $d_{e}$. By inserting edge among depth one element $d_{e}$ we get a chain of depth one element over taxa set $S$. We shall refer to such a chain as a depth one chain in short dchain as shown in Figure 2.11.

| $1,2, A_{2}$ | $3, A_{2}, A_{3}$ <br> (a) Depth one element |  |
| ---: | ---: | ---: |
| $1,2, A_{2}$ | $5,6, A_{1}$ |  |
| $3, A_{2}, A_{3}$ | $4, A_{1}, A_{3}$ | $5,6, A_{1}$ |

(b) Depth one chain

Figure 2.11: Illustration of (a) Depth one element (b) Depth one chain.

### 2.2 Bipartition

Bipartition is a process that takes a pair of taxa set and a quartet set $(S, Q)$ as input. It partitions $S$ into two sets, namely $S_{a}$ and $S_{b}$ with an objective that $\left(S_{a}, S_{b}\right)$ satisfies the maximum number of quartets from $Q$. A bipartition of an unrooted tree $T$ is formed by taking any edge in $T$, and writing down the two sets of taxa that would be formed by deleting that edge. Let $T$ be a tree over the taxa set $S$. Now, if we take an internal edge $e$ of $T$ and delete $e$, then we get two subtrees, namely, $T_{a}$ and $T_{b}$. Let $S_{a}$ and $S_{b}$ be the sets of taxa of $T_{a}$ and $T_{b}$ respectively. We shall denote such bipartition by $\left(S_{a}, S_{b}\right)$. Thus an internal edge in $T$ corresponds to a bipartition of $S$.

### 2.2.1 Method of Bipartition

Bipartition and MaxCut are the state-of-the-art technique to partition the taxa set. But in our research we used bipartition method to partition the taxa set. In conventional divide and conquer based phylogenetic tree construction, the bipartition of the taxa set agrees to an internal edge of the tree under construction. An internal edge, in turn, plays an important role to make quartets to be satisfied or violated against the bipartition. The input of the bipartition algorithm is $(S, Q)$ pair, where $S$ is a set of taxa and $Q$ is a set of quartet. To satisfy the maximum number of quartets from $Q$ we partitions $S$ into two sets, namely, $S_{a}$ and $S_{b}$. The algorithm starts with an initial partition $\left(S_{a_{0}}, S_{b_{0}}\right)$ and iteratively searches for a better partition. Before describing the steps of the algorithm, we describe the following algorithmic ingredients to understand the steps of the bipartition algorithm which is similar to $Q F M$ [1].

Partition Score: We mete the performance of a partition by assigning a partition score. We use a scoring function, score $\left(S_{a}, S_{b}, Q\right)$, as well as high scores indicate a better partition. This function checks each $q \in Q$ against the partition $\left(S_{a}, S_{b}\right)$ and determines whether $q$ is satisfied, violated or deferred. We define the score function in terms of the number of satisfied and violated quartets. Assume that $s$ and $v$ signify the number of satisfied and violated quartets. There are, two natural ways of defining the score function are: (a). taking the difference between the number of satisfied and violated quartets $(s-v)$, and (b). taking the ratio of the number of satisfied and violated quartets $(s / 1+v)$. For the convenience, in this research we used $s-v$ as the score function.

Gain Measure: Let $\left(S_{a}, S_{b}\right)$ be a partition of set of taxa $S$. Let $l \epsilon S$ be a taxon and without loss of generality we assume that $l \epsilon(S-a)$. Let $\left(S_{a^{\prime}}, S_{b^{\prime}}\right)$ be the partition after moving the taxa $l$ from $S_{a}$ to $S_{b}$. That means, $S_{a^{\prime}}=S_{a}-l$, and $S_{b^{\prime}}=S_{b} \cup l$. Then we define the gain of the transfer of the taxon $l$ with respect to $\left(S_{a}, S_{b}\right)$, denoted by Gain $\left(l,\left(S_{a}, S_{b}\right)\right)$, as Score $\left(S_{a^{\prime}}, S_{b^{\prime}}, Q\right)$-Score $\left(S_{a}, S_{b}, Q\right)$.

Singleton Bipartition: A bipartition $\left(S_{a}, S_{b}\right)$ of $S$ is singleton if $\left|S_{a}\right|=1$ or $\left|S_{b}\right|=1$. In our bipartition algorithm, we keep a check for the singleton bipartition. We do not allow our bipartition algorithm to return a singleton bipartition to avoid the risk of an infinite loop.

Algorithm: Now we describe the bipartition algorithm which is known as MFM (Modified FM). Let, $(S, Q)$ be the input to the bipartition algorithm, where $S$ be a set of taxa and $Q$ be a set of quartets over the taxa set $S$. We start with an initial bipartition $\left(S_{a_{0}}, S_{b_{0}}\right)$ of $S$. The initial bipartitioning is done in four steps.

- Step 1: We count the frequency of each distinct quartet in $Q$.
- Step 2: We then sort $Q$ by the frequency count of the quartets in a decreasing order.
- Step 3: Suppose after sorting $Q=\left\{q_{1}, q_{2}, q_{3}, \ldots, q_{k}\right\}$ where $\mathrm{k}=|Q|$. Now we consider the quartets one by one in the sorted order. Initially both $S_{a 0}$ and $S_{b 0}$ are empty.

Let $q \epsilon\left(\left(l_{1}, l_{2}\right),\left(l_{3}, l_{4}\right)\right)$ be a quartet in $Q$. If none of the 4 taxa belongs to either $S_{a 0}$ or $S_{b 0}$, then we insert $l_{1}$ and $l_{2}$ in $S_{a_{0}}$ and $l_{3}$ and $l_{4}$ in $S_{b_{0}}$. Otherwise, if any of the 4 taxa exists
in either $S_{a_{0}}$ or $S_{b_{0}}$ we take the following actions to insert a taxon which doest not exist in $S_{a_{0}}$ or $S_{b_{0}}$. We maintain an insertion order. We consider $l_{1}, l_{2}, l_{3}$ and $l_{4}$ respectively. -To insert $l_{1}$, we look for the partition of $l_{2}$ (if $l_{2}$ exists in any part) and insert $l_{1}$ into that partition. But if $l_{2}$ does not exist in either of the partitions, then we look for the partition of either $l_{3}$ or $l_{4}$ (either of these two must exist in $S_{a_{0}}$ or $S_{b_{0}}$ ) and insert $l_{1}$ into the other partition.
-To insert $l_{2}$, we look for the partition of $l_{1}$ and insert $l_{2}$ into that partition.
-To insert $l_{3}$, we look for the partition of $l_{4}$ (if $l_{4}$ exists in any part) and insert $l_{3}$ into that partition. But if $l_{4}$ does not exist in either of the partitions, then we look for the partition of either $l_{1}$ or $l_{2}$ and insert $l_{3}$ into the other partition.
-To insert $l_{4}$, we look for the partition of $l_{3}$ and insert $l_{4}$ into that partition.

- Step 4: When we insert a taxon $l$ to any part, we remove it from $S$. After considering each $q \epsilon(Q)$ and inserting taxa accordingly, if $S$ remains non-empty, we insert the remaining taxa to either part randomly.

After finding ( $S_{a_{0}}, S_{b_{0}}$ ), we perform iterative search for a better partition. At each iteration, we perform a series of transfers of taxa from one partition set to the other to maximize the number of satisfied quartets. At the beginning of iteration, we set the status of all the taxa as free. Then, for each free taxon $l \epsilon S$, we calculate Gain $\left(l,\left(S_{a_{0}}\right.\right.$, $\left.S_{b_{0}}\right)$ ), and find the taxon $l_{1}$ with the maximum gain. There can be more than one taxa with the maximum gain where we need to break the tie. We will discuss this issue later. Next we transfer $l_{1}$ and set the status of this taxon as locked in the new partition that indicates that it will not be considered to be transferred again in this current iteration. First intermediate bipartition $\left(S_{a_{1}}, S_{b_{1}}\right)$ is formed by this transfer. Similarly the algorithm finds the next free taxon $l_{2}$ with the maximum gain with respect to $\left(S_{a_{1}}\right.$, $S_{b_{1}}$ ), and transfer and lock that taxon to create another intermediate bipartition ( $S_{a_{2}}$, $S_{b_{2}}$ ). Then we transfer all the free taxon one by one in this fashion. Assume that $Q=$


Figure 2.12: An example iteration of the Bipartition Algorithm.
$\{((1,2),(3,4)),((1,2),(5,6)),((1,3),(2,4)),((1,3),(5,6)),((3,4),(5,6)),((2,3),(4,5)),((1,4),(2,5))\}$, be the input quartet set and hence $S=\{1,2,3,4,5,6\}$. Following the steps of the initial bipartition, we get the initial bipartition $S_{a_{0}}=(1,2)$ and $S_{b_{0}}=(3,4,5,6)$. Figure 2.12 shows the first iteration of the bipartition algorithm [1] for this particular example.

Suppose that the taxa are locked in the following order: $\left(l_{1}, l_{2}, l_{3}, \ldots, l_{n}\right)$. That is, $l_{1}$ has been locked first, then $l_{2}, l_{3}$ and so on. Let, the gain values of the corresponding partitions are:

Gain $\left(l_{1},\left(S_{a_{0}}, S_{b_{0}}\right)\right), \ldots, \operatorname{Gain}\left(l_{n},\left(S_{a_{n-1}}, S_{b_{n-1}}\right)\right)$.
Now we define the cumulative gain up to the $k^{\text {th }}$ transfer as
$C \operatorname{Gain}(k)=\sum_{n=1}^{k} \operatorname{Gain}\left(l_{n},\left(S_{a_{i-1}}, S_{b_{i-1}}\right)\right)$
The maximum cumulative gain, MCGain $\left(\left\{l_{1}, l_{2}, l_{3}, \ldots, l_{n}\right\}\right)$ is defined as $\operatorname{MCGain}\left(\left\{l_{1}, l_{2}, l_{3}, \ldots, l_{n}\right\}\right)$
$=\max _{1 \leq i \leq n} C \operatorname{Gain}(i)$
In each iteration, the algorithm finds the current ordering $\left(\left\{l_{1}, l_{2}, l_{3}, \ldots, l_{n}\right\}\right)$ of the transfers

| Step | Taxon | Gain | CGain(k) |
| :---: | ---: | ---: | ---: |
| 1 | 3 | 2 | 2 |
| 2 | 4 | 1 | 3 |
| 3 | 2 | -1 | 2 |
| 4 | 1 | -2 | 0 |
| 5 | 5 | -2 | -2 |
| 6 | 6 | 2 | 0 |

Table 2.1: Gain Summary

| Step | Taxon | Gain | CGain(k) |
| :---: | ---: | ---: | ---: |
| 1 | 4 | 1 | 1 |
| 2 | 2 | -1 | 0 |
| 3 | 1 | -2 | -2 |
| 4 | 5 | -2 | -4 |
| 5 | 6 | 2 | -2 |
| 6 | 3 | 2 | 0 |

Table 2.2: Gain Summary
and saves this order in a log table along with the cumulative gains (see Table 2.1 for example). Let $t_{m}$ be the taxon in the log table corresponding to MCGain ( $\left.\left\{l_{1}, l_{2}, l_{3}, \ldots, l_{n}\right\}\right)$. This means that we obtain the maximum cumulative gain after moving the $m_{t h}$ taxon (with respect to the order stored in the log table). Then we rollback the transfers of the taxa $\left(\left\{l_{m+1}, \ldots, l_{n}\right\}\right)$ that were moved after $l_{m}$. Let the resultant partition after these rollbacks is $\left(S_{a}, S_{b}\right)$. This partition will be the initial partition for the next iteration. In this way, the algorithm continues as long as the maximum cumulative gain is greater than zero and returns the resultant bipartition. Table 2.1 lists the order of locking, corresponding gain and cumulative gain with respect to the iteration illustrated in Figure 2.12. From Table 2.1 we note that we get the maximum cumulative gain 3, after moving taxon 4 . If maximum cumulative gain is tie we break the tie arbitrarily. We consider the taxon for which we get the maximum cumulative gain for the first time. For this example, we get the maximum cumulative gain of 3 at taxon 4 for the first time. So we rollback all the subsequent moves. The resultant partition after this rollback is $(\{1,2,3,4\},\{5,6\})$ (partition $\left(S_{a_{2}}, S_{b_{2}}\right)$ in Figure 2.12). Similarly, Table 2.2 lists the ordering of locking, corresponding gain and cumulative gain with respect to the iteration which follows the iteration illustrated in Figure 2.12. From Table 2.2 we get that the maximum cumulative gain is 1 . So the moves are rolled back and we get the final resultant partition $(\{1,2,3,4\},\{5,6\})$.

As we have mentioned earlier, we do not allow any transfer of taxa that results into a singleton bipartition. Therefore, we need to add some additional conditions. Also, there could be more than one free taxa with the maximum gain, where we need to decide which
one to transfer. We consider the following cases to address these issues. Let, $F$ be a set of free taxa with the maximum gain.

- Case 1: $|F| \geq 1$ and at least one corresponding bipartition is not singleton. That means, there exists $l \epsilon F$ such that transfer of $l$ does not result into a singleton bipartition. Let $F * \subseteq F$ be the set of taxa that can be safely transferred without resulting in a singleton bipartition. Note that, $|F *| \geq 1$. If $|F *|=1$, we transfer the taxa $F *$. Otherwise, we have more than one taxa in $F *$. In that case, we pick the taxon $l \epsilon F *$ for which the corresponding bipartition (after transferring $l$ ) satisfies maximum number of quartets (note that every taxa in $F *$ has the same gain, but the corresponding bipartitions do not necessarily satisfy the same number of quartets). In the case of a tie, we choose one taxon at random.
- Case $2:|k| \geq 1$ and transfer of each $l \epsilon F$ results in a singleton bipartition. In this case, we consider the set of taxa with the second highest maximum gain. Let $F^{\prime}$ be the set of free taxa with the second highest maximum gain. We then recursively check Case 1 and Case 2 on $F^{\prime}$. If we cannot find a taxon that can be transferred without resulting into a singleton bipartition, we make the status of all the free taxa locked and set their gain to zero.

At each divide step we have a $(S, Q)$ pair as input. The bipartition algorithm returns a bipartition $\left(S_{a}, S_{b}\right)$ of the taxa set $S$. We then divide $Q$ into $Q_{a}$ and $Q_{b}$ and obtain $\left(S_{a}, Q_{a}\right)$ and $\left(S_{b}, Q_{b}\right)$ pairs. $S_{a}$ and $S_{b}$ will be further bipartitioned in subsequent divide steps.

### 2.3 Summary

In this chapter we have defined some basic phylogenetic-theoretical terminology related to our research. We have presented bipartition method with a particular example. We also defined the algorithmic ingredients of bipartition method.

## Chapter 3

## Algorithms for Constructing

## MUL-Trees

### 3.1 Algorithms

In this chapter, we study several method to design multi-labeled phylogenetic tree from a set of quartets. We assume that $Q$ is a set of quartets over a set of taxa $S$. We have to construct a MUL-tree from the quartet set $Q$ which satisfied all quartet in $Q$ with minimum number of leaf duplication. In Section 3.1.1 we first propose Randomized divide and conquer approach with randomized duplication technique (RDCRD) which is less effective. In Section 3.1.2 we introduce randomized divide and conquer approach with super-split analysis technique $\left(\mathrm{RDCS}_{\text {Split }} \mathrm{A}\right)$. In Section 3.1.3 we provide a bipartition based divide and conquer approach with randomized duplication technique (BDCRD). In Section 3.1.4 we present a bipartition based divide and conquer approach with super-split analysis technique (QMUL) which is most efficient than above three. Finally in Section 3.1.5 we present another bipartition based divide and conquer approach with quartet analysis technique (AQMUL) which is more efficient and advanced than QMUL.

### 3.1.1 RDCRD Approach

In this section we present randomized divide and conquer approach with randomized duplication technique (RDCRD). RDCRD approach has three steps i) Randomized divide step i) Randomized duplication step and iii) Conquer step.

Randomized Divide: At each recursive step, we partition the taxa set $S$ into two sets ( $S_{a}$ and $S_{b}$ ) randomly. After partitioning the taxa set, the algorithm prolongs both parts ( $S_{a}$ and $S_{b}$ ) with a unique artificial taxon $A_{i}$. This taxon will play an important role while returning from the recursion. We then recurse on both pairs $\left(S_{a}\right)$ and $\left(S_{b}\right)$ until $\left|S_{i}\right| \geq 3$. If $\left|S_{i}\right| \leq 3$, we return depth one element and depth one tree over the taxa set $S$. Finally add another type of dummy taxon $D_{u}$ in any depth one element $d_{e}$ that will help us to merge depth one tree with the caterpillar.

Randomized Duplication: Before duplication process we have perform consistency checking method with respect to depth one chain. By this process we found all inconsistent quartets. We know that each quartet has two parts; the left part and the right part. Now randomly pick left part or right part of each quartet and perform union operation. By this process we get a set of taxa to perform duplication. Now add a dummy taxon $D_{u}$ with the taxa set. Finally we use the taxa set to make a caterpillar. The dummy taxon $D_{u}$ will play an important role to merge depth one tree with the caterpillar.

Conquer: On returning from the recursion, at each step, we have two trees, $T_{a}$ (corresponding to $\left(S_{a}\right)$ ) and $T_{b}$ (corresponding to $\left(S_{b}\right)$ ). These two trees are rerooted at the dummy taxon. Then the dummy taxon is removed from each tree and the two roots are joined by an internal edge. Continue this process until the tree is free from artificial taxon $A_{i}$ and dummy taxon $D_{U}$ and finally get a MUL-tree.

### 3.1.2 RDCS $_{\text {Split }} \mathbf{A}$ Approach

In this section we present randomized divide and conquer approach with super-split analysis technique $\left(\operatorname{RDCS}_{\text {Split }} \mathrm{A}\right) . \mathrm{RDCS}_{\text {Split }} \mathrm{A}$ approach has three steps i) Randomized divide step i)


Figure 3.1: Illustration of RDCRD approach


Figure 3.2: Illustration of $\mathrm{RDCS}_{\text {Split }} \mathrm{A}$ approach

Super-split analysis step and iii) Conquer step.
Randomized Divide: At each recursive step, we partition the taxa set S into two sets ( $S_{a}$ and $S_{b}$ ) randomly. After partitioning the taxa set, the algorithm prolongs both parts ( $S_{a}$ and $S_{b}$ ) with a unique artificial taxon $A_{i}$. This taxon will play an important role while returning from the recursion. We then recurse on both pairs $\left(S_{a}\right)$ and $\left(S_{b}\right)$ until $\left|S_{i}\right| \geq 3$. If $\left|S_{i}\right| \leq 3$, we return depth one element and depth one tree over the taxa set $S$. Finally add another type of dummy taxon $D_{u}$ in any depth one element $d_{e}$ that will help us to merge depth one tree with the caterpillar.

Super-split Analysis: The most crucial part of our algorithm is the super-split analysis technique. It has two parts i) consistency checking method and ii) super-split method. These
two methods will be described in Section 3. After divide step we check the consistency of each quartet by consistency checking process. Finally we get all inconsistent quartets with respect to depth one chain. By super-split method we get expected minimum super-element and it ensures fewer leave duplications. Now add another artificial taxon $D_{u}$ in minimum super-element that will help us to merge the caterpillar with the depth one tree. Consequently we design caterpillar using the taxa from minimum super-element.

Conquer: On returning from the recursion, at each step, we have two trees, $T_{a}$ (corresponding to $\left.\left(S_{a}\right)\right)$ and $T_{b}$ (corresponding to $\left(S_{b}\right)$ ). These two trees are rerooted at the dummy taxon. Then the dummy taxon is removed from each tree and the two roots are joined by an internal edge. Continue this process until the tree is free from artificial taxon $A_{i}$ and dummy taxon $D_{U}$ and finally get a MUL-tree.

### 3.1.3 BDCRD Approach

In this section we present bipartition based divide and conquer approach with randomized duplication technique (BDCRD). BDCRD approach has three steps i) bipartition based divide step ii) Randomized duplication step and iii) Conquer step.

Divide Step: At each divide step, we partition the set of taxa $S$ into two sets $S_{a}$ and $S_{b}$. We shall describe the bipartition algorithm in Section 3. After partitioning the taxa set, the algorithm prolongs both parts ( $S_{a}$ and $S_{b}$ ) with a unique artificial taxon $A_{i}$. The artificial taxon will play an important role while returning from the recursion. After the addition of the artificial taxon to the sets $S_{a}$ and $S_{b}$ we subdivide the quartet set $Q$ into two sets $Q_{a}$ and $Q_{b}$. Now $Q_{a}$ and $Q_{b}$ takes the deferred quartets and the isolated quartets. That means, satisfied and violated quartets with respect to the partition $\left(S_{a}, S_{b}\right)$ are discarded. On the other hand, every deferred quartet where three taxa are in the same part, the other taxon is renamed by the name of the artificial taxon. Then the deferred quartets and the isolated quartets are continues to the next step. Therefore we get, two pairs: $\left(Q_{a}, S_{a}\right)$ and $\left(Q_{b}, S_{b}\right)$. Consequently we recurse on both pairs $\left(Q_{a}, S_{a}\right)$ and $\left(Q_{b}, S_{b}\right)$ until $Q_{i}$ is non-empty and $\left|S_{i}\right| \geq 3$. If either $Q_{i}$ is empty or $\left|S_{i}\right| \leq 3$, we return depth one element $d_{e}$ and depth one tree over the taxa set $S$. Now add

2. Randomized Duplication Step

| $q_{1}:((1,2),(3,4))$ | Quartets |
| :---: | :---: |
| $\left.q_{2}:(1,2),(5,6)\right)$ | $q_{5}:((3,4),(5,6))$ |
| $q_{3}:((1,3),(2,4))$ |  |
| $q_{4}:((1,3),(5,6))$ | $q_{6}:((2,3),(4,5))$ |


Inconsistent Quartets $=((1,3),(2,4)) ;((1,4),(2,5))$
2.2. Union Operation
Duplicated Taxa $=(1,3) \cup(2,5)$
Duplicated Taxa with Dummy Taxon $1,2,3,5, D_{u}$


Figure 3.3: Illustration of BDCRD approach
another type of dummy taxon $D_{u}$ in any depth one element $d_{e}$ that will help us to merge depth one tree with caterpillar.

Randomized Duplication: Before duplication process we have perform consistency checking method with respect to depth one chain. By this process we found all inconsistent quartets. We know that each quartet has two parts; the left part and the right part. Now randomly pick left part or right part of each quartet and perform union operation. By this process we get a set of taxa to perform duplication. Now add a dummy taxon $D_{u}$ with the taxa set. Finally we use the taxa set to make a caterpillar. The dummy taxon $D_{u}$ will play an important role to
merge depth one tree with caterpillar.
Conquer: On returning from the recursion, at each step, we have two trees, $T_{a}$ (corresponding to $\left.\left(S_{a}\right)\right)$ and $T_{b}$ (corresponding to $\left(S_{b}\right)$ ). These two trees are rerooted at the dummy taxon. Then the dummy taxon is removed from each tree and the two roots are joined by an internal edge. Continue this process until the tree is free of dummy taxon and finally get a MUL-tree. Continue this process until the tree is free from artificial taxon $A_{i}$ and dummy taxon $D_{U}$ and finally get a MUL-tree.

### 3.1.4 QMUL Approach

In this section we have present a revolutionary quartet based multi-labeled phylogenetic tree reconstruction algorithm $Q M U L$ (MULtrees from quartets) to construct MUL-trees from quartets with fewer leaf duplications. The QMUL is based on conventional divide and conquer algorithm which has three steps i) Divide step ii) Super-split analysis step and iii) Conquer step. It is based on a bipartition method. Fiduccia et al. in 1982 [9] proposed bipartition method for improving network partitions. Reaz et al. in 2014 modified this method [9] for bipartition of the taxa set we also used this method.

Algorithm. We follow a divide and conquer approach similar to QFM [1]. We here give a very brief description that is essential when we elaborate on the enhancements to the algorithm and the new implementation. Let, $Q$ be a set of quartets over a set of taxa $S$. Our motive is to construct a MUL-tree $T$ on $S$, satisfying all of input quartets. The QMUL approach recursively forms bipartitions of the input set of taxa $S$, where each bipartition corresponds to an internal edge in the tree under construction. QMUL uses a heuristic bipartition technique which is based on finding a maximum cumulative gain of a taxon [1].

Divide: At each recursive step, we partition the taxa set $S$ into two sets $\left(S_{a}\right.$ and $\left.S_{b}\right)$. We have described the bipartitioning algorithm in Method of Bipartition section. After partitioning the taxa set, the algorithm prolongs both parts $\left(S_{a}\right.$ and $\left.S_{b}\right)$ with a unique artificial taxon $A_{i}$. This taxon will play an important role while returning from the recursion. After the addition of the dummy taxon to the sets ( $S_{a}$ and $S_{b}$ ), we subdivide the quartet set $Q$ into two sets, ( $Q_{a}$
and $\left.Q_{b}\right)$. A quartet set $\left(Q_{i}\right.$ takes those quartets $((w, x),(y, z))$ from $Q$ such that either all four taxa $w, x, y$ and $z$ or any three thereof belong to $S_{i}$ (here $i \epsilon(w, x)$ ). That means, satisfied and violated quartets with respect to the partition $\left(S_{a}, S_{b}\right)$ are not considered to be included in either $\left(Q_{a}\right.$ or $\left.Q_{b}\right)$. On the other hand, every deferred quartet, where three taxa are in the same part, the other taxon is renamed by the name of the dummy taxon, and the isolated quartets are continues to the next step. Thus we get, two $\left(Q_{i}, S_{i}\right)$ pairs: $\left(Q_{a}, S_{a}\right)$ and $\left(Q_{b}, S_{b}\right)$. We then recurse on both pairs $\left(Q_{a}, S_{a}\right)$ and $\left(Q_{b}, S_{b}\right)$ until $Q_{i}$ is non-empty and $\left|S_{i}\right|>3$. If either $Q_{i}$ is empty or $\left|S_{i}\right| \leq 3$, we return depth one element $d_{e}$ and depth one tree over the taxa set $S$. Now add another type of dummy taxon $D_{u}$ in any depth one element $d_{e}$ that will help us to merge depth one tree with the caterpillar.

Super-split Analysis: The most crucial part of our algorithm is the super-split analysis technique. It has two parts i) consistency checking method and ii) super-split method. We have described these two methods in Section 3.2. After divide step we check the consistency of each quartet by consistency checking process. Finally we get all inconsistent quartets with respect to depth one chain. By super-split method we get expected minimum super-element and it ensures fewer leave duplications. Now add another artificial taxon $D_{u}$ in minimum super-element that will help us to merge caterpillar tree with depth one tree. Consequently we design caterpillar tree using the taxa from minimum super-element.

Conquer: On returning from the recursion, at each step, we have two trees, $T_{a}$ (corresponding to $\left.\left(S_{a}\right)\right)$ and $T_{b}$ (corresponding to $\left(S_{b}\right)$ ). These two trees are rerooted at the dummy taxon. Then the dummy taxon is removed from each tree and the two roots are joined by an internal edge. Continue this process until the tree is free from artificial taxon $A_{i}$ and dummy taxon $D_{U}$ and finally get a MUL-tree.

Figure 3.4 describes the high level divide and conquer algorithm. Let $Q$ be the input quartet set and $S$ be the corresponding taxa set. Assume that $Q=\{((1,2),(3,4)),((1,2),(5,6))((1,3),(2,4))$, $((1,3),(5,6)),((3,4),(5,6)),((2,3),(4,5)),((1,4),(2,5))\}$, and hence $S=\{1,2,3,4,5,6\}$. By using the bipartition technique initially, $S$ is partitioned into two sets, $S_{a}=\left\{1,2,3,4, A_{1}\right\}$ and $S_{b}=$ $\left\{5,6, A_{1}\right\}$. Here, $A_{1}$ is the artificial taxon. Quartets $q_{2}:((1,2),(5,6)), q_{4}:((1,3),(5,6))$ and


Figure 3.4: Illustration of QMUL approach
$q_{5}:((3,4),(5,6))$, from $Q$ are satisfies with respect to the bipartition $\left(S_{a}, S_{b}\right)$. So these quartets will not be deliberated in the next level. $Q_{a}$ takes $q_{1}:((1,2),(3,4))$ and $q_{3}:((1,3),(2,4))$ as isolated quartets and $q_{6}:((2,3),(4,5))$ and $q_{7}:((1,4),(2,5))$ as deferred quartets. We replace the taxon which does not belong to $S_{a}$ with the artificial taxon $A_{1}$. Hence we get $Q_{a}=\left\{\left((2,3),\left(4, A_{1}\right)\right),\left((1,4),\left(2, A_{1}\right)\right),((1,2),(3,4)),((1,3),(2,4))\right\}$. Similarly we get $Q_{b}=\varnothing$. Next we recurse on $\left(Q_{a}, S_{a}\right)$ and $S_{a}$ partitioned further into $\left(S_{a_{a}}, S_{a_{b}}\right)$. The partition $\left(S_{a_{a}}, S_{a_{b}}\right)$ satisfies $((1,2),(3,4))$, violates $((1,3),(2,4))$ and $\left((1,4),\left(2, A_{1}\right)\right)$ and deferred $\left((2,3),\left(4, A_{1}\right)\right)$, in $Q_{a}$. We replace the taxon which does not belong to $S_{a_{b}}$ with the artificial taxon $A_{2}$. Then we get $Q_{a_{b}}=\left((2,3),\left(4, A_{2}\right)\right)$. Further we recurse on $\left(Q_{a_{b}}, S_{a_{b}}\right)$ and $S_{a_{b}}$ partitioned further into $\left(S_{a_{b_{a}}}, S_{a_{b_{b}}}\right)$. Then we replace the taxon which does not belong to $S_{a_{b_{b}}}$ with the artificial taxon $A_{3}$ and the partition $\left(S_{a_{a_{a}}}, S_{a_{b_{b}}}\right)$ satisfies the quartet $\left((2,3),\left(4, A_{2}\right)\right)$. Whereas the quartet sets for the next level are empty hence we return a depth one element trees for each of the taxa sets $S_{a_{a}}, S_{a_{b_{a}}}, S_{a_{b_{b}}}$, and $S_{b}$. The returned depth one trees trees are merged by removing the artificial taxon of that level and joining the branches of the artificial taxa and we get an intermediate tree with a dummy taxon $D_{u}$. The caterpillar and intermediate tree are merged by removing the dummy taxon $D_{u}$ of that level and joining the branches of the dummy taxon. The conquer steps of Figure 3.4 shows how the trees are returned and merged as the recursion unfolds. Thus we get the final merged tree $(((((1,2), 3), 4), 5), 6),(4,(1,2)))$; this is a MUL-tree satisfying all quartets in $Q$ with three duplicated taxa.

### 3.1.5 Advanced QMUL(AQMUL) Approach

In this section we have presented another revolutionary quartet based multi-labeled phylogenetic tree reconstruction algorithm $Q M U L$ (MULtrees from quartets) to construct MUL-trees from quartets with fewer leaf duplications. The QMUL is based on conventional divide and conquer algorithm which has three steps i) Divide step ii) Quartet Analysis step and iii) Conquer step. It is based on a bipartition method. Fiduccia et al. in 1982 [9] proposed bipartition method for improving network partitions. Reaz et al. in 2014 [9] modified this method for bipartition of the taxa set we also used this method.

2. Quartet Analysis Step

| $q_{1}:((1,2),(3,4))$ | Quartets | $q_{5}:((3,4),(5,6))$ |
| :---: | :---: | :---: |
| $\left.q_{2}:(1,2),(5,6)\right)$ |  | $q_{6}:((2,3),(4,5))$ |
| $q_{3}:((1,3),(2,4))$ | $q_{4}:((1,3),(5,6))$ | $q_{7}:((1,4),(2,5))$ |

2.1. Consistency Checking Method
$d_{\text {chain }} \rightarrow 1,2, A_{2}-3, A_{2}, A_{3}-4, A_{1}, A_{3} \quad 5,6, A_{1}$
Inconsistent Quartets $=((1,3),(2,4)) ;((1,4),(2,5))$
2.2. Quartet Matching Method

| $q-d_{e}$ | $1,2, A_{2}$ | $3, A_{2}, A_{3}$ | $4, A_{1}, A_{3}$ | $5,6, A_{1}$ |
| :---: | :---: | :---: | :---: | :---: |
| $((1,3),(2,4))$ | 1,2 | 3 | 4 |  |
| $((1,4),(2,5))$ | 1,2 |  | 4 | 5 |

Inconsistent Taxa= $\{2\}$
Set of taxa pair with inconsistent taxon $=\{(2,4),(2,5)\}$
Modified $d_{e} \rightarrow \longrightarrow \quad 1,2, A_{2} \quad 3, A_{2}, A_{3} \quad 4, A_{1}, A_{3} \quad(2,5), 6, A_{1}$
3. Conquer Step

Figure 3.5: Illustration of AQMUL approach

Algorithm. We follow a divide and conquer approach similar to QFM [1]. We here give a very brief description that is essential when we elaborate on the enhancements to the algorithm and the new implementation. Let, $Q$ be a set of quartets over a set of taxa $S$. Our motive is to construct a MUL-tree $T$ on $S$, satisfying all of input quartets. The AQMUL approach recursively forms bipartitions of the input set of taxa $S$, where each bipartition corresponds to an internal edge in the tree under construction. AQMUL also used a bipartition technique which is based on finding a maximum cumulative gain of a taxon.

Divide: At each recursive step, we partition the taxa set $S$ into two sets ( $S_{a}$ and $S_{b}$ ). We have already describe the bipartitioning algorithm in Method of Bipartition section. After partitioning the taxa set, the algorithm prolongs both parts ( $S_{a}$ and $S_{b}$,) with a unique artificial
taxon $A_{i}$. This taxon will play an important role while returning from the recursion. After the addition of the dummy taxon to the sets $\left(S_{a}\right)$ and $\left(S_{b}\right)$, we subdivide the quartet set $Q$ into two sets, $\left(Q_{a}\right)$ and $\left(Q_{b}\right)$. A quartet set $\left(Q_{i}\right)$ takes those quartets $((w, x),(y, z))$ from $Q$ such that either all four taxa $w, x, y$ and $z$ or any three thereof belong to $S_{i}$ (here $i \epsilon(w, x)$ ). That means, satisfied and violated quartets with respect to the partition $\left(S_{a}, S_{b}\right)$ are not considered to be included in either $\left(Q_{a}\right.$ or $\left.Q_{b}\right)$. On the other hand, every deferred quartet, where three taxa are in the same part, the other taxon is renamed by the name of the dummy taxon, and the isolated quartets are continues to the next step. Thus we get, two $\left(Q_{i}, S_{i}\right)$ pairs: $\left(Q_{a}, S_{a}\right)$ and $\left(Q_{b}, S_{b}\right)$. We then recurse on both pairs $\left(Q_{a}, S_{a}\right)$ and ( $\left.Q_{b}, S_{b}\right)$ until $Q_{i}$ is non-empty and $\left|S_{i}\right|>3$. If either $Q_{i}$ is empty or $\left|S_{i}\right| \leq 3$, we return depth one element $d_{e}$ and depth one tree over the taxa set $S_{i}$. Now add another type of dummy taxon $D_{u}$ in any depth one element $d_{e}$ that will help us to merge depth one tree with the caterpillar.

Quartet Analysis Technique: The most crucial part of $A Q M U L$ algorithm is the quartet analysis technique. It has two parts i) consistency checking method and ii) quartet matching method. We have described these two methods in Section 3.3. After divide step we check the consistency of each quartet by consistency checking process. Finally we get all inconsistent quartets with respect to depth one chain. By quartet matching method we get inconsistent taxon list and expected taxa pairs. Each taxa pair contain an inconsistent taxon and a normal taxon. Now we have to make same taxa pairs in depth one elements by adding inconsistent taxon in depth one elements and separate that pairs from other pairs, taxa or taxon by first bracket.

Conquer: On returning from the recursion, at each step, we have two trees, $T_{a}$ (corresponding to $\left(S_{a}\right)$ ) and $T_{b}$ (corresponding to $\left(S_{b}\right)$ ). These two trees are rerooted at the dummy taxon. Then the dummy taxon is removed from each tree and the two roots are joined by an internal edge. Continue this process until the tree is free from artificial taxon $A_{i}$ and dummy taxon $D_{U}$ and finally get a MUL-tree.

Figure 3.5 describes the high level divide and conquer algorithm. Let $Q$ be the input quartet set and $S$ be the corresponding taxa set. Assume that $Q=\{((1,2),(3,4)),((1,2),(5,6))((1,3),(2,4))$,
$((1,3),(5,6)),((3,4),(5,6)),((2,3),(4,5)),((1,4),(2,5))\}$, and hence $S=\{1,2,3,4,5,6\}$. By using the bipartition technique initially, $S$ is partitioned into two sets, $S_{a}=\left\{1,2,3,4, A_{1}\right\}$ and $S_{b}=$ $\left\{5,6, A_{1}\right\}$. Here, $A_{1}$ is the artificial taxon. Quartets $q_{2}:((1,2),(5,6)), q_{4}:((1,3),(5,6))$ and $q_{5}:((3,4),(5,6))$, from $Q$ are satisfies with respect to the bipartition $\left(S_{a}, S_{b}\right)$. So these quartets will not be deliberated in the next level. $Q_{a}$ takes $q_{1}:((1,2),(3,4))$ and $q_{3}:((1,3),(2,4))$ as isolated quartets and $q_{6}:((2,3),(4,5))$ and $q_{7}:((1,4),(2,5))$ as deferred quartets. We replace the taxon which does not belong to $S_{a}$ with the artificial taxon $A_{1}$. Hence we get $Q_{a}=\left\{\left((2,3),\left(4, A_{1}\right)\right),\left((1,4),\left(2, A_{1}\right)\right),((1,2),(3,4)),((1,3),(2,4))\right\}$. Similarly we get $Q_{b}=\varnothing$. Next we recurse on $\left(Q_{a}, S_{a}\right)$ and $S_{a}$ partitioned further into $\left(S_{a_{a}}, S_{a_{b}}\right)$. The partition $\left(S_{a_{a}}, S_{a_{b}}\right)$ satisfies $((1,2),(3,4))$, violates $((1,3),(2,4))$ and $\left((1,4),\left(2, A_{1}\right)\right)$ and deferred $\left((2,3),\left(4, A_{1}\right)\right)$, in $Q_{a}$. We replace the taxon which does not belong to $S_{a_{b}}$ with the artificial taxon $A_{2}$. Then we get $Q_{a_{b}}\left((2,3),\left(4, A_{2}\right)\right)$. Further we recurse on $\left(Q_{a_{b}}, S_{a_{b}}\right)$ and $S_{a_{b}}$ partitioned further into $\left(S_{a_{b_{a}}}, S_{a_{b_{b}}}\right)$. Then we replace the taxon which does not belong to $S_{a_{b_{b}}}$ with the artificial taxon $A_{3}$ and the partition $\left(S_{a_{a_{a}}}, S_{a_{b_{b}}}\right)$ satisfies the quartet $\left((2,3),\left(4, A_{2}\right)\right)$. Whereas the quartet sets for the next level are empty hence we return a depth one element trees for each of the taxa sets $S_{a_{a}}, S_{a_{b_{a}}}, S_{a_{b_{b}}}$, and $S_{b}$. Then we apply quartet analysis technique and we get all inconsistent quartets with respect to depth one chain. Here $((1,3),(2,4))$ and $((1,4),(2,5))$ are inconsistent quartets. By quartet matching method we get inconsistent taxon list and expected taxa pairs. The inconsistent taxon list $=\{2\}$ and expected taxa pairs $=(2,5)$. Now add the inconsistent taxon 2 in depth one element with the normal taxon 5 to make the taxa pair $(2,5)$. After the modification of the depth one elements the algorithm returns depth one trees for each of the taxa sets $S_{a_{a}}, S_{a_{b_{a}}}, S_{a_{b_{b}}}$, and $S_{b}$. The returned trees are merged by removing the artificial taxon of that level and joining the branches of the artificial taxa. The conquer steps of Figure 3.5 shows how the trees are returned and merged as the recursion unfolds. Thus we get the final merged tree $((((1,2), 3), 4),(5,2), 6)$; this is a MUL-tree which satisfied all quartets in $Q$ with one duplicated taxon.

(b) Depth one chain

Figure 3.6: Consistency of (a) quartets with respect to (b) depth one chain.

### 3.2 Super-split Analysis Technique

In this section, we study super-split analysis technique which is the most decisive part of our algorithms. It has two parts i) consistency checking method and ii) super-split method. In Section 3.3.1 we first present consistency checking method. In Section 3.3.2 we introduce super-split method.

### 3.2.1 Consistency Checking Method

A quartet $((1,2),(3,4))$ is consistent with respect to depth one chain if the path from 1 to 2 does not overlap the path from 3 to 4 in depth one chain. In Figure 3.6 among the three quartets, quartet $q_{1}$ is consistent with respect to dchain as there exists a non-overlapping path from 1 and 2 to 3 and 4. Other two quartets are inconsistent with respect to dchain as no such path exists in dchain.

Algorithm: Now we describe the consistency checking algorithm which we call $C C$ algorithm. Let, $Q$ be the input to the consistency checking algorithm, where $Q$ be a set of quartets over the taxa set $S$. Consistency checking are performed by the following five steps:

- Step 1: Take all quartets set $Q$ as input.
- Step 2: Now pick each quartet $q_{i} \epsilon Q$ one by one and check its consistency with respect to depth one chain.
- Step 3: A quartet $q_{i} \epsilon Q$ is inconsistent with respect to depth one chain if the path from first taxa to second taxa overlaps the path from third taxa to fourth taxa of a quartet;

| Input quartets |  |  |
| :---: | :---: | :---: |
| $q_{1}:((1,2),(3,4))$ | $q_{3}:((1,3),(2,4))$ | $q_{5}:((3,4),(5,6))$ |
| $q_{2}:((1,2),(5,6))$ | $q_{4}:((1,3),(5,6))$ | $q_{6}:((2,3),(4,5))$ |
|  | $q_{7}:((1,4),(2,5))$ |  |
| $1,2, A_{2}$ | Depth one chain |  |
| $Q_{i}=\{((1,3),(2,4)),(((1,4),(2,5))\}$ |  |  |

Figure 3.7: Illustration of consistency checking method.
otherwise the quartet is consistent with respect to depth one chain.

- Step 4: Discard all the consistent quartets from the input list.
- Step 5: Finally list all inconsistent quartets $Q_{i n}$.

Figure 3.7 describes the consistency checking method. Let $Q=((1,2),(3,4)),((1,2),(5,6))$, $((1,3),(2,4)),((1,3),(5,6)),((3,4),(5,6)),((2,3),(4,5)),((1,4),(2,5))$. After applying consistency checking algorithm we get inconsistent quartet set $Q_{\text {in }}=\{((1,3),(2,4)),(((1,4),(2,5))\}$.

### 3.2.2 Super-split Method

The most crucial part of our algorithm is the super-split method. Super-split method returns some super-elements with the minimum number of leaves. The super-elements with the minimum number of leaves ensure fewer leave duplications. From the minimum super-elements we construct a caterpillar.

Algorithm: Let, $\left(S_{i n}, Q_{i n}\right)$ be the input to the super-split method, where $Q_{i n}$ be a set of inconsistent quartets over the taxa set $S_{i n}$ that we found from quartet checking method. We perform super-split method over inconsistent quartets $Q_{i n}$ as follows:

- Step 1: Take an inconsistent quartet $Q_{i n}$ as a root.
- Step 2: Insert two edges to the left side and right side to the root.
- Step 3: Take the next inconsistent quartet from $Q_{i n}$ and put it along left edge; reverse the quartet and put it along right edge.


Figure 3.8: Illustration of super-split method.

- Step 4: Now take union of root and left edge value and it will serve as a root for next step.
- Step 5: Also take union of root and right edge value and it will serve as a root for next step.
- Step 6: Continue this process until $Q_{i n}$ is empty.
- Step 7: Finally we get a binary tree of super-split and found all minimum super-elements with minimum number of taxa.

Figure 3.8 describes the super-split method. Assume that inconsistent quartet set $Q_{\text {in }}=$ $((1,3),(2,4)) ;(((1,4),(2,5))$. After applying super-split method we have three minimum superelements $\{1,3,4\},\{2,4,5\}$ and $\{1,2,4\}$.

### 3.3 Quartet Analysis Technique

In this section, we study quartet analysis technique with a particular example. It is the most important part of AQMUL algorithm. It has two parts i) consistency checking method and ii) quartet matching method. Recall Subsection 3.2.1 for consistency checking method. In Section 3.3.2 we introduce quartet matching method.

### 3.3.1 Consistency Checking Method

In AQMUL approach we have used the consistency checking method to find all inconsistent quartet $Q_{i n}$ in $Q$ with respect to depth one chain. It is also an important part of AQMUL approach. Recall Section 3.2 for consistency checking method.

### 3.3.2 Quartet Matching Method

The most crucial part of quartet analysis technique is the quartet matching method. Quartet Matching method returns minimum number of inconsistent taxa to perform duplication.

Algorithm: Let, $\left(Q_{i n}, d_{e}\right)$ be the input to the quartet matching method, where $Q_{i n}$ be a set of inconsistent quartets and $d_{e}$ be the depth one elements which we get from divide steps. We perform quartet matching method over inconsistent quartets $Q_{i n}$ and depth one elements $d_{e}$ as follows:

- Step 1: Take all depth one element and put all depth one element on a table in separate column.
- Step 2: Select an inconsistent quartet and put every taxon of each pair of a quartet on next row, bellow the depth one element.
- Step 3: Find the taxon which makes the quartet inconsistent and also find the taxa pair which contain the inconsistent taxon. If any quartet have more than one inconsistent taxon then select any one of them randomly.
- Step 4: Now select the next quartet and put every taxon of each quartet bellow depth one element. Now continue the step 3 and find the expected taxa pair. To choose the inconsistent taxon we give priority of previous inconsistent taxa.
- Step 5: Continue this process until inconsistent quartet list is empty.
- Step 6: Finally quartet matching method returns a list of inconsistent taxa and a list of taxa pair where each pair contain exactly one inconsistent taxon. If the list contains

Quartet Matching Table

| $Q_{i} d_{e}$ | $1,2, A_{2}$ | $3, A_{2}, A_{3}$ | $4, A_{1}, A_{3}$ | $5,6, A_{1}$ |
| :--- | :--- | :---: | :---: | :---: |
| $((1,3),(2,4))$ | 1,2 | 3 | 4 |  |
| $((1,4),(2,5))$ | 1,2 |  | 4 | 5 |

Inconsistent Taxa=\{2\}
Set of taxa pair with inconsistent taxon $=\{(2,4),(2,5)\}$
Modified depth one element
$1,2, A_{2} \quad 3, A_{2}, A_{3} \quad 4, A_{1}, A_{3} \quad(2,5), 6, A_{1}$

Figure 3.9: Illustration of quartet matching method.
more than one pair which contain same inconsistent taxon then we take the pair which has maximum pair length.

Figure 3.9 describes the quartet matching method. Assume that inconsistent quartet set $Q_{i n}=((1,3),(2,4)) ;(((1,4),(2,5))$. After applying quartet matching method we have one inconsistent taxon 2 and the set of taxa pair with inconsistent taxon $=\{(2,4),(2,5)\}$.

### 3.4 Summary

In this chapter we have proposed some algorithms; randomized divide and conquer approach with randomized duplication technique ( RDCRD ), bipartition based divide and conquer approach with randomized duplication technique (BDCRD), randomized divide and conquer approach with super-split analysis technique $\left(\mathrm{RDCS}_{\text {Split }} \mathrm{A}\right)$ bipartition based divide and conquer approach with super-split analysis technique (QMUL) and bipartition based divide and conquer approach with quartet analysis technique (AQMUL). In this chapter, we study super-split analysis technique and quartet analysis technique which are the most crucial part of our algorithms.

## Chapter 4

## Performance of Algorithms

This chapter provides the simulation and simulation results on synthetic and real datasets. We also analyze the simulation results to find out the performance of our algorithms. In this thesis we mainly have proposed two algorithms QMUL and AQMUL to construct MUL-trees from a set of quartets. We have analyzed the performance of QMUL approach and AQMUL approach in terms of percentage of duplications and compare with QFM approach. We also have proposed three auxiliary techniques to design MUL-trees and compare our methods with auxiliary techniques in terms of average number of duplications. The auxiliary techniques are following:

1. Randomized divide and conquer approach with randomized duplication technique (RDCRD).
2. Bipartition based divide and conquer approach with randomized duplication technique (BDCRD).
3. Randomized divide and conquer approach with super-split analysis technique ( $\left.\operatorname{RDCS}_{\text {Split }} \mathrm{A}\right)$.

First, the experimental setup for the simulations and experiments on simulated data to design MUL-trees are presented. In the second section of the chapter experiments on real data and results on real data are presented.

### 4.1 Experiments on Simulated Datasets

We perform simulation on synthetic and real datasets. For all data, the QMUL and AQMUL algorithms was run on a computer with a 3.2 GHz core i7 processor and 8GB RAM. Here we perform simulation of QMUL and AQMUL algorithms with other three auxiliary techniques to design MUL-trees. In order to test the performance of QMUL and AQMUL algorithms, we conducted experiments on synthetic dataset and real datasets and generate 15 model MUL-trees under different conditions. The properties we wanted to measure are the number of duplication and the running times of our methods.

### 4.1.1 Simulated Datasets

To scrutinize the performance of our method on various model conditions, we have generated quartet sets, from a set of model trees, by varying the number of taxa ( $n$ ), the number of quartets $(q)$ and the percentage of consistent quartets $(c)$ with respect to the model tree $(90 \%$ consistency level means that $10 \%$ quartets are flipped to disagree with the model tree). Specifically, we flipped either $10 \%$ or $30 \%$ of the quartets (yielding 90 and 70 percent correct quartets, respectively). We have generated model species trees with $n=25,50,100,200$ and 300 taxa. To generate the model trees and the input quartet sets, we have used the tool developed and used in [9]. The tool takes as input the number of taxa $(n)$, number of quartets $(q)$ and the consistency level $(c)$, and returns the quartet sets accordingly. For $n=25,50,100,200,300$ we have generated $n^{1.25}, n^{1.5}$ and $n^{2}$ quartets. We have not generated more quartets because $n^{2}$ quartets have been empirically shown to be enough to construct very accurate phylogenetic trees. Although $n^{1.25}$ is a small number, we have chosen this size to test the performance of both methods on a comparatively smaller number of quartets as well. For each size $(q)$, we have varied the percentage of consistent quartets (c) by making it $70 \%$ and $90 \%$.

Table 4.1: Results on synthetic dataset [1, 4] in terms of average number of duplications.

| $n$ | $q$ | $\mathrm{C}=70 \%$ |  |  |  |  | $\mathrm{C}=90 \%$ |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | RDCRD | RDCSSplitA | BDCRD | QMUL | AQMUL | RDCRD | RDCSSplitA | BDCRD | QMUL | AQMUL |
| 25 | 50 | 24 | 24 | 23 | 19 | 7 | 21 | 20 | 15 | 8 | 5 |
| 25 | 125 | 24 | 23 | 24 | 20 | 15 | 22 | 23 | 17 | 15 | 10 |
| 25 | 625 | 24 | 24 | 22 | 21 | 18 | 23 | 22 | 20 | 18 | 17 |
| 50 | 133 | 43 | 42 | 40 | 32 | 13 | 40 | 40 | 42 | 30 | 9 |
| 50 | 354 | 44 | 42 | 41 | 41 | 30 | 41 | 42 | 40 | 32 | 20 |
| 50 | 2500 | 47 | 45 | 43 | 43 | 35 | 45 | 47 | 44 | 40 | 35 |
| 100 | 317 | 93 | 90 | 87 | 72 | 28 | 86 | 80 | 76 | 70 | 25 |
| 100 | 1000 | 92 | 91 | 90 | 93 | 61 | 88 | 83 | 81 | 71 | 54 |
| 100 | 10000 | 97 | 94 | 92 | 94 | 85 | 90 | 85 | 84 | 85 | 71 |
| 200 | 753 | 190 | 191 | 185 | 144 | 98 | 190 | 189 | 189 | 158 | 88 |
| 200 | 2829 | 193 | 190 | 190 | 158 | 144 | 191 | 191 | 190 | 158 | 128 |
| 200 | 40000 | 193 | 193 | 191 | 162 | 150 | 192 | 191 | 191 | 162 | 150 |
| 300 | 1249 | 285 | 281 | 278 | 231 | 144 | 284 | 281 | 278 | 213 | 120 |
| 300 | 5197 | 290 | 285 | 281 | 240 | 228 | 290 | 288 | 281 | 220 | 210 |
| 300 | 90000 | 297 | 295 | 296 | 261 | 240 | 295 | 291 | 289 | 245 | 234 |

### 4.1.2 Simulated Data Results

For a phylogeny reconstruction algorithm, if a certain tree or network is used to obtain the input data, the algorithm should return exactly this tree or network. This is an important property for reconstructing phylogenies and known as the consistency principle. We now present the results on the simulated datasets mentioned above. In each case, we have compared the average number of duplication for the trees estimated by the RDCRD, RDC $S_{\text {split }} \mathrm{A}, \mathrm{BDCRD}, \mathrm{QMUL}$ and AQMUL. The results for $\mathrm{c}=70 \%$, and $\mathrm{c}=90 \%$ are summarized in Table 4.1. Figure 4.1 shows the bar charts comparing the values presented in Table 4.1. The results in Table 4.1 are presented in batches for different values of $n$ as follows. For $n=25,50,100,200$ and 300 we have three rows, one each for $q=n^{1.25}, q=n^{1.5}$ and $q=n^{2}$. The top most row of each batch of Table 4.1 shows the results when $q={ }^{n 1: 5}$ (from left to right, the consistency levels reported are $70 \%$ and $90 \%$, respectively). From the Table 4.1 we see that for $80 \%$ of our data sets the average number of duplications are $\mathrm{RDCRD}>\mathrm{RDC} S_{\text {split }} \mathrm{A}>\mathrm{BDCRD}$. For the remaining $20 \%$ of the data the number of duplications of $\operatorname{RDCRD}, \mathrm{RDC} S_{S p l i t} \mathrm{~A}$ and BDCRD are fluctuating. But the
performance of the QMUL and the AQMUL is always better than the RDCRD, RDCS split A and BDCRD. If we compare the performance of QMUL and AQMUL according to the results presented in Table 4.1, it is clear that the average number of duplications in AQMUL approach always smaller than the average number of duplications in QMUL approach. That means AQMUL produces better trees than QMUL.


Figure 4.1: A chart showing the difference in performance among five techniques in terms of number of duplications when $\mathrm{c}=90 \%$ and $\mathrm{q}=n^{1} .25$.

In 2014 Reaz et al. [1] apply QFM method on similar data sets and construct supertree. Snir et al. [4] in 2008 also use these data sets to construct supertree by applying the QMC method. Now we show the comparison of QFM method with our method. The results are summarized in Table 4.2. In Table 4.2 CQ indicates percentages of consistent quartet (CQ\%), IQ indicate percentages of inconsistent quartet (IQ\%) and DR indicate percentages of duplication (DR\%). Here Table 4.2 shows that the QFM constructed supertree from $70 \%$ and $90 \%$ consistent quartets without duplications. But the supertree constructed by QFM cannot ensure the consistency of each quartet. The first row of each batch of Table 4.2 shows the results with $n^{1.25}$ quartets.

Table 4.2: Comparison of QFM, QMUL and AQMUL under various model conditions [1].

| $n$ | $q$ | QFM |  |  |  | QMUL |  |  |  | AQMUL |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | $\mathrm{c}=70 \%$ |  | $\mathrm{c}=90 \%$ |  | $\mathrm{c}=70 \%$ |  | c=90\% |  | c=70\% |  | c=90\% |  |
|  |  | CQ | IQ | CQ | IQ | CQ | DR | CQ | DR | CQ | DR | CQ | DR |
| 25 | 50 | 62 | 38 | 70 | 30 | 100 | 76 | 100 | 32 | 100 | 28 | 100 | 20 |
| 25 | 125 | 50 | 50 | 73 | 27 | 100 | 80 | 100 | 60 | 100 | 60 | 100 | 40 |
| 25 | 625 | 65 | 35 | 85 | 15 | 100 | 84 | 100 | 72 | 100 | 72 | 100 | 68 |
| 50 | 133 | 65 | 35 | 70 | 30 | 100 | 64 | 100 | 60 | 100 | 26 | 100 | 18 |
| 50 | 354 | 50 | 50 | 73 | 27 | 100 | 82 | 100 | 64 | 100 | 60 | 100 | 40 |
| 50 | 2500 | 70 | 30 | 90 | 10 | 100 | 86 | 100 | 80 | 100 | 70 | 100 | 70 |
| 100 | 317 | 65 | 35 | 71 | 29 | 100 | 72 | 100 | 70 | 100 | 28 | 100 | 25 |
| 100 | 1000 | 55 | 45 | 70 | 30 | 100 | 93 | 100 | 71 | 100 | 61 | 100 | 54 |
| 100 | 10000 | 71 | 29 | 91 | 9 | 100 | 94 | 100 | 85 | 100 | 85 | 100 | 71 |
| 200 | 753 | 65 | 35 | 66 | 34 | 100 | 72 | 100 | 79 | 100 | 49 | 100 | 44 |
| 200 | 2829 | 61 | 39 | 85 | 15 | 100 | 79 | 100 | 79 | 100 | 72 | 100 | 64 |
| 200 | 40000 | 69 | 31 | 93 | 7 | 100 | 81 | 100 | 81 | 100 | 75 | 100 | 75 |
| 300 | 1249 | 70 | 30 | 74 | 26 | 100 | 77 | 100 | 71 | 100 | 48 | 100 | 40 |
| 300 | 5197 | 75 | 25 | 79 | 21 | 100 | 80 | 100 | 73 | 100 | 76 | 100 | 70 |
| 300 | 90000 | 72 | 28 | 91 | 9 | 100 | 87 | 100 | 82 | 100 | 80 | 100 | 78 |

When $q=n^{1.25}$ and the consistency of quartets is $70 \%$ the QFM can ensure the consistency of $62 \%$ quartets. So QFM method lost $38 \%$ quartets. But the AQMUL and the QMUL ensure the consistency of $100 \%$ quartets by allowing $28 \%$ and $76 \%$ duplications respectably. In case of $90 \%$ consistent quartets and $q=n^{1.25}$ the QFM can ensure the consistency of only $70 \%$ quartets and losing $30 \%$ quartets. In this stage the AQMUL and the QMUL requires $20 \%$ and $32 \%$ duplications respectably. The second row of each batch of Table 4.2 shows the results with $n^{1.5}$ quartets. For $q=n^{1.5}$ and $\mathrm{C}=70 \%$ the consistency of QFM is $50 \%$ and the loss of QFM is $50 \%$. In this case the AQMUL and the QMUL ensure the consistency of $100 \%$ quartets by allowing $60 \%$ and $80 \%$ duplications respectably. In case of $90 \%$ consistent quartets and


Figure 4.2: Comparison of QMUL and AQMUL in terms of number of duplications under various model conditions.
$q=n^{1.5}$ the QFM can ensure the consistency of only $73 \%$ quartets and losing $27 \%$ quartets. In this stage the AQMUL and the QMUL requires $40 \%$ and $60 \%$ duplications respectably. The third row of each batch of Table 4.2 shows the results with $n^{2}$ quartets. When $q=n^{2}$ and $\mathrm{C}=70 \%$ the QFM reconstruct highly accurate species trees and can ensure the consistency of approximately $65 \%$ to $72 \%$ quartets. When $q=n^{2}$ and $\mathrm{C}=90 \%$ the QFM reconstruct highly accurate species trees and can ensure the consistency of approximately $85 \%$ to $93 \%$ quartets. But for $q=n^{2}$ and $\mathrm{C}=70 \%$ both AQMUL and QMUL requires average $76 \%$ and $86 \%$ leaf duplications reasonably and for $\mathrm{c}=90 \%$ both AQMUL and QMUL requires average $72 \%$ and $80 \%$ leaf duplications respectably. With $q=n^{2}$ quartets, both QMUL and AQMUL begin to produce MUL-trees which duplication is larger than that of $q=n^{1.25}$ and $q=n^{1.5}$ quartets.

Figure 4.2 show percentages of duplication of QMUL and AQMUL on the simulated datasets.

Table 4.3: Comparison of QFM, QMUL and AQMUL under the conflict-free model (100\% consistent quartets) conditions [1].

| $n$ | $q$ | QFM |  | QMUL |  | AQMUL |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | CQ | IQ | CQ | DR | CQ | DR |
| 25 | 125 | 100 | 0 | 100 | 0 | 100 | 0 |
| 50 | 1000 | 100 | 0 | 100 | 0 | 100 | 0 |
| 100 | 600 | 100 | 0 | 100 | 0 | 100 | 0 |
| 200 | 753 | 100 | 0 | 100 | 0 | 100 | 0 |
| 300 | 1249 | 100 | 0 | 100 | 0 | 100 | 0 |

We show average number of duplications (over 15 replicates of data) for each model condition. We varied the number of taxa $(n)$, number of quartets $(q)$ and the percentage of consistency level $(c)$. For a particular value of $q$ and $c$ the number of taxa is varied along the X -axis, the average number of duplications is shown along the Y-axis. From left to right: the number of quartets are $n^{1.25}, n^{1.5}$ and $n^{2}$. From top to bottom: $70 \%$ and $90 \%$ of the input quartets are consistent with the model species tree.

We have also differentiated the performance of QFM, QMUL and AQMUL under the conflict-free model conditions in terms of percentages of duplication. The experimental results are shown in Table 4.3. In Table 4.3 CQ indicates percentages of consistent quartet (CQ\%), IQ indicate percentages of inconsistent quartet (IQ\%) and DR indicate percentages of duplication (DR\%). From the table, we can say that the performance of QFM, QMUL and AQMUL under the conflict-free model conditions are same. That means when the consistency level is $100 \%$ the QMUL and the AQMUL construct the same tree as the tree constructed by the QFM. The table 4.3 shows that when the consistency level is $100 \%$ the percentages of duplication of QMUL and AQMUL are zero. We have also evaluated the running time of QFM, QMUL and AQMUL under the conflict-free model conditions. From the experiments we can say that the running time of QFM, QMUL and AQMUL under the conflict-free model conditions are same.

### 4.1.3 Computational Issues

We have evaluated the running time and memory usage of QMUL and AQMUL. On smaller datasets, both QMUL and AQMUL run in few seconds. For example, on 25 taxa, QMUL took
between 3 minutes to 4 minutes (depending on the number of quartets), and AQMUL took less than 3 minutes. However, QMUL is much slower than AQMUL on the larger datasets. For example, QMUL took 17 hours for the largest datasets of our experiment with 300 taxa and 40,000 quartets, while AQMUL took only 12 hours. From our experiments we have seen that if the number of inconsistent quartets are increasing the computational time of QMUL increase exponentially but computational time of AQMUL is not increase exponentially. We believe that this difference is due to the naive implementation of our algorithm. AQMUL has been implemented in a very efficient code, and it scales well on larger datasets. We are currently working on improving our implementation using advanced data structures. We have also measured the memory usage by these methods. QMUL and AQMUL are memory efficient and use only few megabytes of memory. For example, the peak memory usages by QMUL on the datasets with 300 taxa and 90,000 quartets are 45 MB and the peak memory usages by AQMUL on the datasets with 300 taxa and 90,000 quartets are 44 MB .

### 4.2 Experiments on Real Datasets

In this section we now present the results on the real datasets. In each case, we have compared the AQMUL with the QMUL based on the average number of duplications. Depending on the number of duplications we also compare our methods with MTRT method [6].

### 4.2.1 Real Datasets

To test the performance of the QMUL and AQMUL on real biological datasets, we applied both QMUL and AQMUL on two real datasets. The first dataset containing high-polyploid North American and Hawaiian violets [31]. All major morphological groups occurring in North America were sampled. All sequence were aligned with MUSCLE [32] and phylogenies were constructed using maximum likelihood. The second dataset dataset containing the flowering plant genus Silene (Caryophyllaceae) consisting of 12 plants was published in [30]. The gene trees in [30] are reconstructed using standard techniques in phylogenetic analysis from regions of


Figure 4.3: An original MUL-tree on flowering plants with 7 duplications.


Figure 4.4: The obtained MUL-tree by applying AQMUL on the quartets extracted from the MUL-tree shown in Figure 4.3. This MUL-tree has 7 duplications.
the nuclear RNA polymerase gene family, two concatenated chloroplast regions and one nuclear ribosomal region, see [12] for more details. We collect the real biological datasets and real MULtree from [6]. We have decomposed the original MUL-trees into its induced quartets which is called embedded quartets. Then, we have taken the union of all these quartets (multiple copies of a quartet have been retained). The labels represent Silene species, namely, S.ajanensis $(A)$, S. uralensis $(U)$, S. involucrata $(I)$, S. sorensenis $(S)$, S. ostenfeldii $(O)$, S. zawadskii $(Z)$, S. linnaeana $(L)$, S. uralensis (Mongolia) $(U M)$, S. samojedora (SAM), S. villosula $(V)$, S.sachalinensis (SAC) and S. tolmatchevii ( $T$ ).

### 4.2.2 Real Data Results

We have used the real quartets to estimate a MUL-tree by using both QMUL and AQMUL. The AQMUL construct MUL-trees which has less or equal number of duplications than that of the original MUL-tree. In 2014 Hassanzadeh et al. [6] construct MUL-trees by using triplet based method (MTRT) from rooted triplets. They also used the gene trees in [30, 31] and extracted all triplets and then apply MTRT on these triplets.

Table 4.4: Performance analysis of MTRT approach, QMUL approach and AQMUL approach on real datasets $[6,30,31]$.

| Real MUL-Trees | MTRT | QMUL | AQMUL |
| :---: | :---: | :---: | :---: |
| MUL-tree on flowering plants <br> with 7 duplications | 5 | 10 | 7 |
| MUL-tree on violet species <br> with 20 duplications | 18 | 27 | 20 |

The original MUL-tree for first dataset has 7 duplications, whereas the MUL-tree produced by the MTRT has 5 duplications, the QMUL has 10 duplications and the AQMUL has 7 duplications. Figure 4.3 and Figure 4.4 show the original MUL-tree and the MUL-tree constructed by AQMUL respectively. The original MUL-trees for second datasets have 20 duplications, whereas the MUL-trees produced by AQMUL and QMUL have 20 and 27 duplications respectively. The MUL-tree produced by the MTRT has also have 18 duplications for second data set. Figure 4.5 and Figure 4.6 show the original MUL-tree and the MUL-tree constructed by


Figure 4.5: An original MUL-tree on violet species with 20 duplications.


Figure 4.6: The obtained MUL-tree by applying AQMUL on the quartets extracted from the MUL-tree shown in Figure 4.5. This MUL-tree has 20 duplications.

AQMUL respectively. Table 4.4 show the performance of MTRT approach, QMUL approach and AQMUL approach on two real datasets.


Figure 4.7: A chart showing the difference in performance among MTRT, QMUL and AQMUL on real datasets in terms of number of duplications.

Figure 4.7 show number of duplications of MTRT, QMUL and AQMUL on the real datasets. The average number of duplications was shown along the Y-axis. The real gene trees are marked along the X -axis.

### 4.3 Duplication Vs Consistency

It has been proved that the most frequently occurring quartet (on a set of four taxa) in the gene trees is a statistically consistent estimator of the species tree on this set of four taxa. Therefore, estimating a species tree by maximizing the number of consistent quartets is a statistically consistent approach for arbitrarily large numbers of genes. In the presence of gene tree discordance and estimation error, it may not be possible to estimate a species tree which is consistent to all the quartets in a gene tree. However, researchers can construct MUL-trees by allowing arbitrarily large number of leaf duplications to ensure the consistency of all quartets. But constructing a MUL-tree by keeping the number of duplication to a minimum is very challenging.

There is no mathematical, biological or empirical evidence that minimizing the number of duplications will guarantee statistical consistency . That means, inferring a MUL-tree with the minimum number of leaf duplications may not converge to the correct MUL-tree. But, since in many cases, parsimonious approaches which aim to construct a tree by minimizing number of duplication and losses or deep coalescences or horizontal gene transfers may produce good estimates of evolutionary trees. Therefore, constructing MUL-trees by minimizing leaf duplications and maximizing satisfying quartets should be a reasonable approach.

The algorithm that we have proposed in this dissertation is not an exact algorithm, i.e, it does not guarantee the minimum numbers of leaf duplications. In such cases, we can consider a trade-off between the numbers consistent quartets and the number of leaf duplications. We can think of sacrificing some quartets in order to keep the number of duplications within a reasonable limit. We leave this as a future work to investigate this trade-off by conducting an extensive simulation study.

### 4.4 Summary

In this chapter we have showed simulation and experimental performance of algorithms on synthetic data sets and real data sets. The computational issues of the algorithms also presented in this chapter. We have provided comparison of QFM, QMUL and AQMUL approach under various model conditions. Finally from the results of the simulation we have proved that the experimental performance of the AQMUL approach is better than the QMUL approach. The supporting information of these simulations are provided in the appendix.

## Chapter 5

## Conclusion

In this thesis we have studied the problem to construct MUL-trees from a set of quartets. We have given some algorithms for this problem and provided a simulation of these algorithms. The goal of the algorithm is to construct MUL-trees with fewer leaf duplication that is consistent with the input set of quartets and minimizes the number of its duplications. All the algorithms was implemented in $\mathrm{C} \#$. In this thesis we have also dealt with different theoretical aspects of MUL-trees. We now briefly describe the content of each chapter of this thesis.

We have started with an introductory overview on phylogenetic tree, quartet based phylogenetic tree inference and MUL-trees in Chapter 1. In that chapter we have given our problem definition, described our objective of this thesis and presented its motivation. We have provided some applications of MUL-trees like perusing host-parasite cospeciation, constructing gene trees, constructing phylogenetic networks and in the study of biogeography. We have introduced the similar problems found in the literature. Finally we have provided the results of this thesis and its organization.

In Chapter 2 we have introduced the preliminary ideas on MUL-trees, supertrees and phylogenetic network. We have also discussed bipartition method, define split and super-split. Depth one chain is also described in this chapter.

In Chapter 3 we have resolved the problem to construct MUL-trees from a set of quartets. In this chapter we have also discussed about the algorithms for MUL-tree construction. In this
chapter we have given some algorithms (RDCRD approach, $\mathrm{RDCS}_{\text {Split }} \mathrm{A}$ approach, BDCRD approach, QMUL approach and AQMUL approach) for constructing MUL-trees. We have also introduced super-split analysis technique and quartet analysis technique which is the most crucial part of our algorithms.

Finally in Chapter 4 we have conducted a simulation to see the practical performance of our algorithms on synthetic and real datasets. Finally we have showed that the performance of AQMUL is better than other four techniques that we proposed in this thesis. The code of this simulation is provided in the appendix.

We now address different important and challenging problems regarding MUL-trees construction. We now present some open problems and future research scopes related to this thesis.

1. Note that a phylogenetic network can be associated to a MUL-tree. Therefore, it seems that constructing the MUL-tree from a set of quartets could be an alternative method for the problem of constructing a phylogenetic network with minimum reticulation from a set of quartets. In future work it could be interesting to explore ways to also speed-up the super-split analysis technique of the QMUL approach, which will ensure the consistency of each quartet in a set with minimum number of duplications.
2. To design an efficient super-split method to construct the super-split with minimum super elements and keep the computation time minimum.
3. To design more efficient quartet analysis technique which will play an important role to ensure the consistency of each quartet in a set with minimum number of duplications. Because the efficiency of AQMUL approach mainly depends on efficient quartet analysis technique.
4. To study the complexity of the problem and also analysis the parameterized complexity of the problem.
5. To set a parameter in AQMUL and QMUL approach to count the number of duplications by using these parameter.
6. As the performance of AQMUL and QMUL approach decrease with the increase of quartet size $\left(q \geq n^{2}\right)$ so there is a scope to modify the AQMUL approach.
7. Execute theoretical analysis of the problem to construct MUL-trees with fewest leaf duplication.

## Appendix A

## Supporting Information

The QMUL and AQMUL approach was implemented in C\# and is freely available for use from http://cse.buet.ac.bd/research/group/gd/index.php?pageid=Resources.htm

## List of Publications

1. Rahman, M. M., and Rahman, M. S., Multi-labeled Phylogenetic Tree Reconstruction from Quartets, Poster presented at: Workshop on Bioinformatics and Stringology (BioS), 2015.
2. Rahman, M. M., and Rahman, M. S., Algorithms for Constructing Multi-labeled Phylogenetic Trees from Quartets with Fewer Leaf Duplications, Proc. of The International Conference on Bioinformatics and Biostatistics for Agriculture, Health and Environment (ICBBAHE2017), pp. 55, 2017.

## References

[1] Reaz, R., Bayzid, M. S. and Rahman, M. S., Accurate Phylogenetic Tree Reconstruction from Quartets: A Heuristic Approach, PLOS ONE, Vol 9(8): e104008, 2014.
[2] Snir, S., Warnow, T. and Rao, S., Short Quartet Puzzling: A New Quartet-Based Phylogeny Reconstruction Algorithm, Journal of Computational Biology, Vol. 15, No.1, pp. 91103, 2008.
[3] Page, D. M., Modified Mincut Supertrees, Proc. of Workshop on Algorithms in Bioinformatics, Lecture Notes in Computer Science, Vol. 2452, Springer, pp. 537-551, 2002.
[4] Snir, S. and Rao, S., Quartets MaxCut: A Divide and Conquer Quartets Algorithm, EEE/ACM Transaction of Computational Biology and Bioinformatics, Vol. 7, No. 04, pp. 704718, 2010.
[5] Alon, N., Snir, S. and Yuster, R., On the Compatibility of Quartet Trees, SIMA Journal on Discrete Mathematics, Vol. 28, No. 3, pp. 14931507, 2014.
[6] Hassanzadeh, R., Eslahchi, C. and Sung, W. K., Do Triplets Have Enough Information to Construct the Multi-Labeled Phylogenetic Tree? PLOS ONE, Vol.9, pp.1-10, 2014.
[7] Poormohammadi, H., Eslahchi, C. and Tusserkani, R., TripNet: A Method for Constructing Rooted Phylogenetic Networks from Rooted Triplets, PLOS ONE, Vol.9, pp.1-12, 2014.
[8] Huber, K. T., Lott, M., Moulton, V. and Spillner, A., The Complexity of Deriving MultiLabeled Trees from Bipartitions, Journal of Computational Biology, Vol. 15, No. 6, pp. 639-51, 2008.
[9] Fiduccia, C. M., and Mattheyses, R. M., A Linear-Time Heuristic for Improving Network Partitions, Proc. of the 19th Design Automation Conference, Vol. 0146-7123, pp.175181, 1982.
[10] Semple, C., and Steel, M., Phylogenetics, Oxford University Press, Oxford, UK, 2003.
[11] Steel, M., The Complexity of Reconstructing Trees from Qualitative Characters and Subtrees, J.Classification, Vol. 9, pp. 91116, 1992.
[12] Huber, K. T., Oxelman, B., Lott, M., and Moulton, V., Reconstructing the Evolutionary History of Polyploids from Multi-Labeled Trees, Molecular Biology and Evolution, Vol. 23(9), pp. 17841791, 2006.
[13] Page, R. D. M., and Charleston, M. A., Trees within trees: phylogeny and historical associations, Trends in Ecology and Evolution, Vol. 13(9), pp. 356359, 1998.
[14] Brown, G.K., Nelson, G. and Ladiges, P. Y., Historical biogeography of Rhododendron section Vireya and the Malesian Archipelago, Journal of Biogeography, Vol. 33, pp. 19291944, 2006.
[15] Snir, S., and Rao, S., Quartet maxcut: A fast algorithm for amalgamating quartet trees, Journal of Molecular Phylogenetics and Evolution, Vol. 62, pp. 18, 2012.
[16] Lott, M., Spillner, A., Huber K. T., and Moulton, V., PADRE: a package for analyzing and displaying reticulate evolution, Bioinformatics, Vol. 25(9), pp. 11991200, 2009.
[17] Berry, V., Jiang, T., Kearney, P., Li, M., and Wareham, T., Quartet Cleaning: Improved Algorithms and Simulations, European Symposium on Algorithms, Vol. 1643, pp. 313324, 1999.
[18] Ganapathy, G., Goodson, B., and Jansen, R., Pattern Identification in Biogeography, IEEE/ACM Transaction of Computational Biology and Bioinformatics, Vol. 3, pp. 334346, 2006.
[19] Page, R.D.M., Parasites, Phylogeny and Cospeciation, Int. J. Parasitol. Vol. 23, 499506, 1993.
[20] Guillemot, S., Jansson, J., and Sung, W. K., Computing a Smallest Multi-labeled Phylogenetic Tree from Rooted Triplets, IEEE/ACM Transactions on Computational Biology and Bioinformatics, Vol. 8, No. 04, pp. 1141-1147, 2011.
[21] Deepak, A., Baca, D. F., and McMahon, M. M., Extracting Conflict-Free Information from Multi-labeled Trees, Algorithms for Molecular Biology, Vol. 8 No. 18, pp. 8-18, 2013.
[22] Asai, T., Arimura, H., Uno, T., and Nakano, S., Discovering Frequent Substructures in Large Unordered Trees, Discovery Science, pp. 47-61, Springer, 2003.
[23] Chou, S., and Hsu, C. L., MMDT: A Multi-Valued and Multi-Labeled Decision Tree Classifier for Data Mining, Expert Systems with Applications, vol. 28, pp. 799-812, 2005.
[24] Crochemore, M., and Verin, R., Direct Construction of Compact Directed Acyclic Word Graphs, Proc. Ann. Symp. Combinatorial Pattern Matching, pp. 116-129, 1997.
[25] Cotton J. A., and Page, R. D. M., Rates and Patterns of Gene Duplication and Loss in the Human Genome, Royal Society of London, Biological science, editor, Proceedings, vol. 272, pp. 277283, 2005.
[26] Linder, C. R., and Warnow, T., An Overview of Phylogeny Reconstruction, Book chapter, in S. Aluru (editor), Handbook of Computational Biology, Chapman and Hall, CRC Computer and Information Science Series, Vol. 324, pp. 1561-1564, 2005.
[27] Huson, D. H., Rupp, R., and Scornavacca, C., Phylogenetic Networks: Concepts, Algorithms and Applications, Cambridge University Press, 2010.
[28] Bruce Rannala, B., Huelsenbeck, J. P., Yang, Z., and Nielsen, R., Taxon Sampling and the Accuracy of Large Phylogenies, Systematic Biology, vol. 47, pp. 702-710, 1998.
[29] Eulenstein, O., Chen, D., Burleigh, J. G., Baca, D. F., and Sanderson, M. J., Performance of Flip Supertrees with a Heuristic Algorithm, Systematic Biology, vol. 53, no. 2, pp. 299308, 2004.
[30] Popp M, Erixon P, Eggens F, Oxelman B, Origin and Evolution of a Circumpolar Polyploid Species Complex in Silene (Caryophyllaceae) Inferred from Low Copy Nuclear RNA Polymerase Introns, rDNA, and Chloroplast DNA, Systematic botany 30(2): 302313, 2005.
[31] Marcussen, T., Jakobsen, K. S., Danihelka, J., Ballard, H. E., Blaxland, K., Brysting, A. K., Oxelman, B., Inferring Species Networks from Gene Trees in High-Polyploid North American and Hawaiian Violets (Viola, Violaceae), Systematic biology, Vol. 61(1). pp. 107126, 2012.
[32] Edgar R. C., MUSCLE: Multiple Sequence Alignment with High Accuracy and High Throughput, Nucleic acids research, Vol. 32(5), pp. 17921797, 2006.
[33] Bryanta, D., and Lagergrenb, J., Compatibility of unrooted phylogenetic trees is FPT, Theoretical Computer Science, Vol. 351, pp. 296 302, 2006.
[34] Guillemot, S., Jansson, J., and Sung, W. K., Computing a Smallest Multilabeled Phylogenetic Tree from Rooted Triplets, IEEE/ACM Transactions on Computational Biology an Bioinformatics, Vol. 8, No. 4, pp. 1141-1147, 2011.

## Index

bifurcating, 19
biogeography, 5
biomedical science, 5
caterpillar, 21
cladograms, 5
consistency, 1, 13, 64
consistent, 25
deferred, 25
depth one chain, 26
depth one element, 26
depth one tree, 26
DNA, 3
evolutionary biology, 1
evolutionary events, 24
evolutionary tree, 17
gene evolution, 5
gene tree, 5,22
inconsistent pair, 25
inconsistent taxon, 25
isolated, 25

Maximum Quartet Consistency, 8

Method of Bipartition, 39, 43
MFM, 28
minimum super-element, 26
molecular systematic, 5
MUL-tree, 33, 34, 37, 39, 40, 42, 44, 45
Multi-labeled Phylogenetic Trees, 2
multi-labeled tree, 21
multifurcating, 19

NP hard, 2
number of duplications, 22, 64
phylogenetic network, 23
phylogenetic networks, 5
phylogenetic tree, 1, 17
Phylogenetics, 17
phylogeny, 3
polyploid species, 5

QFM, 27
quartet, 24
quartet analysis, 13
resolved, 24
rooted bifurcating tree, 19
rooted multi-labeled phylogenetic tree, 21
rooted networks, 23
rooted phylogenetic tree, 18
satisfied, 24
species tree, 22
split, 25
super-split, 26
super-split analysis, 13
supertree, 1, 20
tree, 16
child, 17
internal node, 17
leaf, 17
nodes, 17
parent, 17
root, 17
rooted tree, 17
triplet, 24
unresolved, 24
unrooted bifurcating tree, 19
unrooted MUL-tree, 2
unrooted multi-labeled phylogenetic tree, 21
unrooted phylogenetic network, 23
unrooted phylogenetic tree, 18
unrooted tree, 18
violated, 25

