Phylogenetic Tree Construction Using Chemical Reaction Optimization

Avijit Bhattacharjee

Student ID: 140212

Sk Rahad Mannan

Student ID: 140214

Computer Science and Engineering Discipline

Khulna University

Khulna-9208, Bangladesh

August, 2018.

Khulna University

Computer Science and Engineering Discipline

The undersigned hereby certify that they have read and recommend to the Computer Science and Engineering Discipline for acceptance of a thesis entitled "**Phylogenetic Tree Construction Using Chemical Reaction Optimization**" by **Avijit Bhattacharjee, Sk Rahad Mannan** in partial fulfillment of the requirements for the degree of **Bachelor of Science in Computer Science and Engineering (CSE).**

Dated: 14th August, 2018

Dr. Md. Rafiqul Islam Thesis Supervisor
Professor
Computer Science and Engineering Discipline
Khulna University
Khulna.

Dr. Kazi Shah Nawaz Ripon Second Examiner
Professor
Computer Science and Engineering Discipline
Khulna University
Khulna.

Dr. Md. Anisur Rahman Head of the Discipline
Professor
Computer Science and Engineering Discipline
Khulna University
Khulna.

Acknowledgement

First of all, we would like to thank Almighty Allah for giving us enough mental and physical strength to complete our thesis properly. We would like to express our sincere gratitude to our supervisor Dr. Md. Rafiqul Islam, Professor, Computer Science and Engineering Discipline, Khulna University for his cooperation, suggestion, guidance and continuous encouragement through the course of the study.

We are highly grateful to our second examiner Dr. Kazi Shah Nawaz Ripon, Professor, Computer Science and Engineering Discipline, Khulna University for reviewing this report and giving us suggestions to improve it.

Besides our supervisor and second examiner we would like to acknowledge our honorable teachers of computer Science and Engineering Discipline for their encouraging support and discussion. We also thank our parents for their encouragement, support, and attention.

We are also thankful to our classmates for their moral support which helped us to accomplish our thesis.

Abstract

Phylogenetic tree construction (PT) problem is a well-known NP-hard optimization problem that finds most accurate tree representing evolutionary relationships among species. Different criteria are used to measure the quality of a phylogeny tree by analyzing their relationships and nucleotide sequences. With increasing number of species, solution space of phylogenetic tree construction problem grows exponentially. As NP-hard problems are both resource and time consuming, the optimal solutions for those problems are unknown in polynomial time. On the other hand, approximation algorithms try to give approximate optimal solutions to such NP-hard problems in polynomial time. The approximate solutions can be obtained by state-of-the-art, heuristic and meta-heuristic approaches. In this thesis, we have implemented Chemical Reaction Optimization algorithm to solve phylogeny construction problem for multiple datasets. For exploring both local and global search, we have implemented four elementary operators of CRO for phylogeny construction problem. Two-correction methods have been designed for finding good combination of species according to maximum likelihood and maximum parsimony criteria. Our approach is compared with artificial bee colony, genetic algorithm, and meta-population genetic algorithm. The experimental results show that for maximum parsimony criterion our implemented algorithm gives better results for three real datasets and same for one dataset. On the other hand, for maximum likelihood criterion our proposed algorithm gives better results for two datasets among four datasets.

Table of Content

Chapter

List of Figures

List of Tables

Chapter I

Background

1.1 Introduction

Phylogeny is a visual demonstration showing the genetic affiliation or evolutionary relationship among distinct taxa [1]. A phylogenetic tree is the study of genealogical links between interrelated organisms in evolutionary history [2]. It can be rooted or unrooted. In a computational scenario, phylogenies are binary unrooted trees. Branching patterns of phylogenies represent the evolution of species, which form successions of common ancestors. The aligned entities are located at the tips where common ancestors represent the inner nodes. The correlation between taxa denoted as an edge or branch. The length of a branch represents the mutation time [2]. Phylogenetic tree enriches our understanding capability of how genes, genomes, species (and molecular sequences more generally) are evolved. Phylogenies give us an insight of how species change in history and how the species will change in future. It is important in forensics, getting assess of DNA evidence, to identify the pathogen origin, in conservation policy and in Bioinformatics [3].

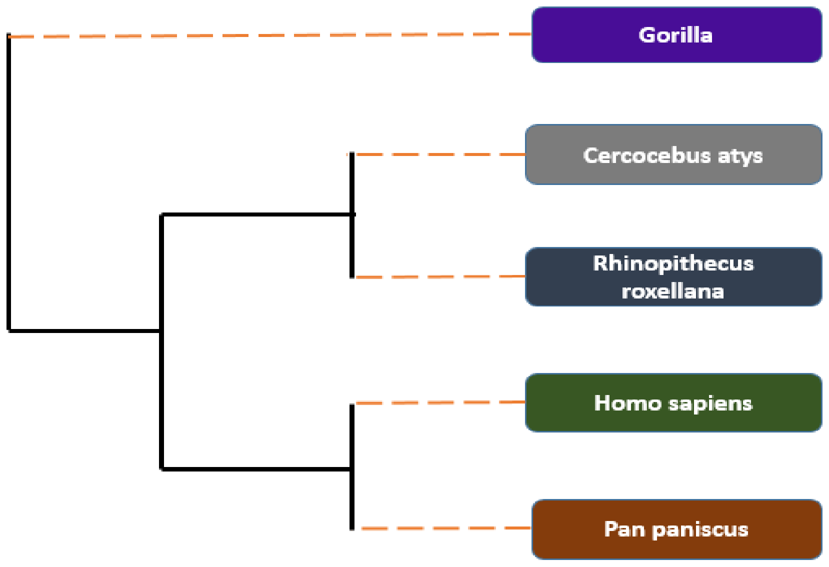

Fig. 1 Phylogenetic tree.

For example, Figure 1 is a phylogenetic tree, which is constructed using human (*Homo sapiens*) myoglobin protein (P02144) sequence of length 154. For getting another four species myoglobin, protein sequence BLAST [4-5] search is done. We have selected five species to construct the tree. Then we have used MetaPIGA [4] for constructing this tree. In this phylogenetic tree, the correlation between taxa is denoted by the branches. The species are placed to their nearly close characteristics species to form a phylogeny inference.

Phylogenies are used to evaluate DNA evidence, to identify the origin of pathogens and molecular sequencing. Phylogeny approaches can also be used to know about a new pathogen outbreak. It discovers how species are related to and subsequently the likely source of transmission. This can lead to a new suggestion for public health policy [3]. Phylogenies can be applied in forensics (dental practice in HIV transmission). In conservation biology, phylogenies help to prevent extinction (illegal whale hunting). Other applications of phylogenies include multiple sequence alignment, protein structure prediction, gene and protein function prediction, epidemiology, drug development and drug design [3].

There are several techniques for phylogenetic tree inference. Some techniques proceed with matching DNA, RNA or Protein sequences and group similar species together (node) and join them with edges, which denote estimated time for evolution. Another way is to generate all types of possible topologies and then minimize candidate for an optimal solution with a metaheuristic algorithm. Phylogenetic tree construction methods are divided into two categories [5]. They are algorithmic methods and optimality criterion methods. Algorithmic methods reach to the final solution without judging all solutions in the search space and these are the quick producer of solutions. Algorithmic methods construct a tree using distance base matrix. Distance matrix examples are Neighbor-Joining (NJ), Unweighted Pair Group Method for Arithmetic Mean (UPGMA). Another type is optimality criterion based, which works by optimizing an objective function and deploying a search technique. In this process, objective function calculates the score of each tree in search space and find the best-scored solution. Example for optimality criterion methods are maximum parsimony and maximum likelihood. In order to get better maximum parsimony score trees e.g. PHYLIP[6] and PAUP [7] are used and for better maximum likelihood score trees e.g. fastDNAml [8], PHYML [9], RAxML [10] are used.

In this thesis, we have implemented an algorithm based on the chemical reaction optimization algorithm to solve the phylogenetic tree construction problem. Our main target is to find out a combination of species, which describes evolution history almost properly. The chemical reaction is a procedure involving the reorganization of the molecular and ionic structure of substances. There is a common nature of this universe that every molecule or ion that is not in the stable state wants to be stable by chemical reaction. The important feature of CRO is its capability of searching. It has both local and global search properties. These two properties make CRO perfect for any optimization problem and finding out the global best results of the optimization problems. In recent years, CRO has successfully solved many optimization problems and showed better results than other metaheuristics approaches. We have used four reaction operators of CRO: on-wall ineffective collision, decomposition, inter-molecular ineffective collision and synthesis to find the global optimal point. These operators make the whole process able to search the combination of species, which explains proper evolution. We have implemented two extra operators: reform1 and reform2. Reform1 validates the solution of synthesis operator and reform2 improves the final ML scores.

1.2 Motivation

Phylogenetic tree construction is very important in the study evolutionary process. The good evolutionary tree is necessary in biological studies. Exact algorithms take a large amount of time and space. For better performance, we have looked for a metaheuristic algorithm. Chemical reaction algorithm is a powerful metaheuristic algorithm. The main power of CRO algorithm is searching both local and global solution space. Another important characteristic is reaction operators. They are: on-wall ineffective collision, decomposition, inter-molecular ineffective collision and synthesis. We have used above mentioned four CRO operators and two extra operators to find a good phylogenetic tree. In recent years, CRO has successfully solved many optimization problems such as quadratic assignment problem[11], resource-constrained project scheduling problem [11], channel assignment problem in wireless mesh networks [11], grid scheduling problem, stock portfolio selection problem [11], artificial neural network training [11], network coding optimization problem [11]. Therefore, recent successes of CRO have motivated us for applying it in phylogenetic tree construction problem.

1.3 Objective of the Thesis

We have proposed an approach based on chemical reaction optimization to solve the Phylogenetic Tree Construction problem. We have used the basic four operators of CRO and two repair operators. Two repair operators are reform1 and reform2. The purposes of this thesis are given below:

- To compare the performance of our proposed method with the existing algorithms.
- To redesign the operators of CRO with respect to the phylogenetic tree construction problem.
- To get good trees according to maximum parsimony and maximum likelihood score.

1.4 Organization of the Thesis

There are six chapters in this thesis. The content of each chapter has been shortly given below.

Chapter I (Background): This chapter contains the introductory concepts and brief overview of the phylogenetic tree construction problem. It also includes application areas, motivation, and objectives of this thesis.

Chapter II (Phylogenetic Inference): This chapter explains the basic concepts of phylogenetics, complexity and objective functions briefly.

Chapter III (Literature Review): This chapter presents brief discussion of different existing exact, heuristics and meta-heuristics algorithms for the phylogenetic tree construction problem.

Chapter IV (Chemical Reaction Optimization for Phylogenetic Tree Construction Problem): In this chapter, we briefly discuss about population generation, solution representation, different reaction operators, reform functions and parameter settings of our proposed algorithms.

Chapter V (Experimental results and comparison with other algorithms): This chapter demonstrates comparison of the experimental results of the proposed methods with MOABC, TNT, DNAPARS for maximum parsimony and MOABC, GARLI, RAxML METAPIGA for maximum likelihood.

Chapter VI (Conclusion and Future Directions): In this chapter, we conclude our thesis along with limitations and some future research directions.

Chapter II

Phylogenetic Inference

2.1 Intoduction

In this section, we have presented the basic concept of phylogenetics, the complexity of phylogenetic tree construction problem and two optimality criteria which are the scope of this research. They are maximum parsimony and maximum likelihood.

2.2 Basic Concepts of Phylogenetics

The phylogenetic tree represents the evolutionary history of ethnological relationships among linked entities. By statistical and algorithmic procedures, phylogenetic methods have found the most correct assumption about the evolution of taxa. We consider the input is an alignment taken by n sequences of m characters. For DNA sequences, A, C, G, T that represents Adenine, Cytosine, Guanine, and Thymine respectively define site values. The sites also represent the nitrogenous base. The output of phylogeny inference is a tree shape structure. Let T = (V, E) be a phylogenetic tree generated from a set of n sequences of m nucleotides, u, v $\in$ V two related nodes, (u, v) $\in$ E the ancestral connection between both nodes and u_i, v_i the state at the ith site on the sequence for u and v.

2.3 Complexity of the problem

Construction of phylogenies is a very complex computation task because for 50 species there exists 2.68×10^{76} number of topologies, which is almost near the number of atoms in the universe [3]. Phylogenetic tree construction problem is an NP-hard problem [2]. For greater values of n (e.g. organism), no polynomial time solution exists. However, the importance of a good algorithm for phylogenetic tree construction is so extreme that some large projects are underway, such as CIPRES (Cyber Infrastructure for Phylogenetic Research www.phylo.org) and ATOL (Assembling the Tree of Life project, tolweb.org). In order to assess the complexity of these problem two things need to be tackled [12]. Firstly, if the number of sites increases, then the computational cost of objective function will increase. Secondly, if the number of taxa increases in our dataset, then the space of all possible tree grows exponentially. For a unrooted tree, if we have n taxa, then equation 1 tells the possible number of phylogenetic topologies [2].

$$\frac{(2n-5)!}{(n-3)!2^{n-3}} \tag{1}$$

2.4 Maximum Parsimony criterion

Maximum parsimony is the process that minimizes total number of character state changes and mutation facts. It is the simplest process to construct the good phylogenetic tree. Let T = (V, E) be a phylogenetic tree generated from a set of n sequences of m nucleotides. Here u, v $\in$ V two related nodes and (u, v) $\in$ E the ancestral connection between both nodes. In equation 2, u_i, v_i the state at the ith site on the sequence for u and v. The parsimony score of T [13, 14]:

$$\mathrm{P(T)} = \sum_{\mathrm{i=1}}^{m} \sum_{(u,v)\in E} C_i(u,v), \tag{2}$$

Where the cost value C_i(u, v) is defined by comparing the character states u_i and v_i as follows: If $u_i \neq v_i$, then C_i(u, v) will be one. Otherwise, C_i(u, v) will be zero. In this process, a bifurcating tree is constructed with an additive process. This process gradually takes species and group them together. In the first step, trees have to be traversed from tips to root. A tip node move forward its intermediate node that is linked with another tip node. If two tips node have same state, then intermediate node takes common state. If tips node have different state, then intermediate node takes both state with logical "or". MP calculation algorithm reaches to the root node following above procedure. In the tree of Figure 2, node 7 has the state C or G because node 6 and node 5 have different states. Similarly, node 8 has A or G state. Node 9 has state G as node 7, 8 have common state G. Next, node 10 has state G or C because node 2 and node 9 have different states. By this procedure, MP calculation have reached to root node 1. Now, MP calculation starts from root to tips traversal for counting mutations. In this traversal, MP solves the contradictions created by tips to root traversal. If parent node's state and child node's state differ, then anyone of the "or" state can be selected. Otherwise, common state between parent and child node is placed as child nodes state. Root node 1 and its child node 10 has no common states so contradiction at node 10 is solved by putting anyone of the "or" state. In this case, node 10 takse state G. The branch

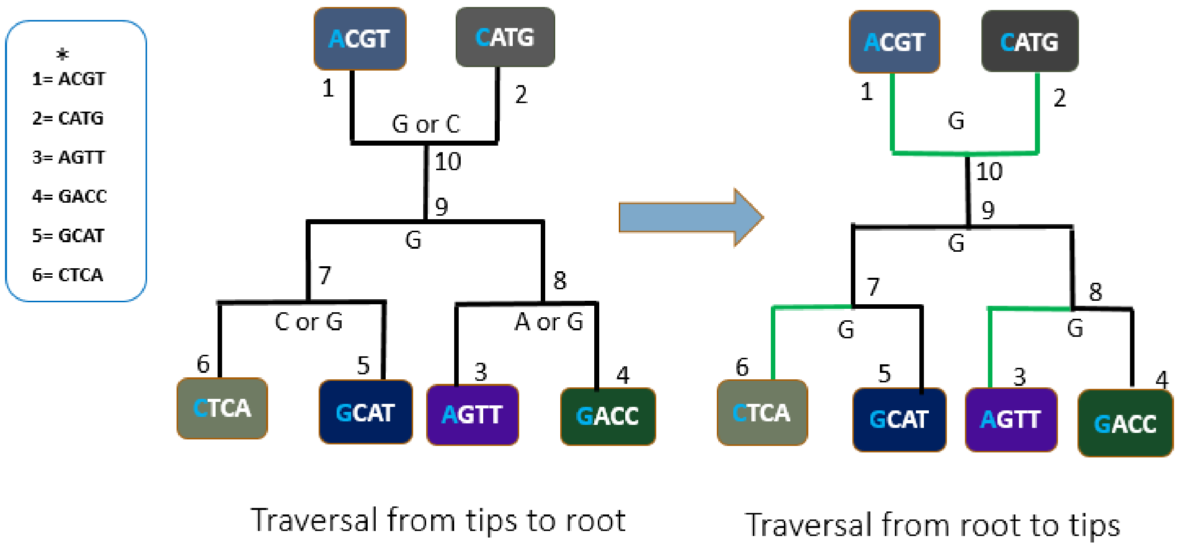

Fig. 2 Maximum Parsimony

connecting node 1 and node 10 has change in state. So, this branch has one parsimony score. Similarly, the branch connecting node 2 and node 10 also has one parsimony score. Node 9 has no contradiction. So, the traversal moves forward to node 7 and node 8. As parent of node 7 and node 8 have common state G, contradiction of these two nodes are solved by keeping state G. The branch connecting node 6 and node 7 have different state, so it has one parsimony score. Similarly, the branch connecting node 3 and node 8 also have one parsimony score. Total parsimony score for first column site is 4. This is the value of inner loop in equation 2. The outer loop sums up parsimony score for all four column site which is in this case 16.

2.5 Maximum Likelihood

Maximum likelihood value is the probability of observing a tree being correct evolutionary relationship among a set of species under some specific models. There are many evolutionary models. To select a model, sequences are analyzed. Models define the probability of nucleotide base transitions. etc. There are 16 possible transition states for nucleotide bases such as $A \rightarrow G$, $G \rightarrow A$, $G \rightarrow T$, $C \rightarrow T$.

For example, a tree consisting four species 1, 2, 3, 4 are shown in Figure 3. Here, X, Y are two hypothetical species sequence. However, their sequence is unknown. For any site X, Y can have either A, C, G, T. Equation 3 calculates any of the combinations of X, Y values probability of being at that site. Now all the probability values for X, Y at a specific site is summed up as equation 4. Value of equation 4 is the likelihood value of a specific site, not for the whole sequence of data. To get likelihood, a probability value for each site using equation 3 and equation 4 is calculated. Equation 5 multiplies each site likelihood values from equation 4. Here P_{XY} is the transition probability of base X to become base Y according to the selected model. Moreover, P_{xy} is the value for any pair of nucleotide base for a specific site. Here x, y can take A, C, G, T bases. P_i is the likelihood probability for each site. There are four sites in the given example. Equation 3, 4 and 5 are taken from [15].

$$\mathrm{P}_{xy} = \mathrm{P}_{\mathrm{X}} \times P_{XC} \times P_{XG} \times P_{XY} \times P_{YT} \times P_{YT} \tag{3}$$

$$P_i = \sum P_{xy} \tag{4}$$

$$P = \prod_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{P_i} \tag{5}$$

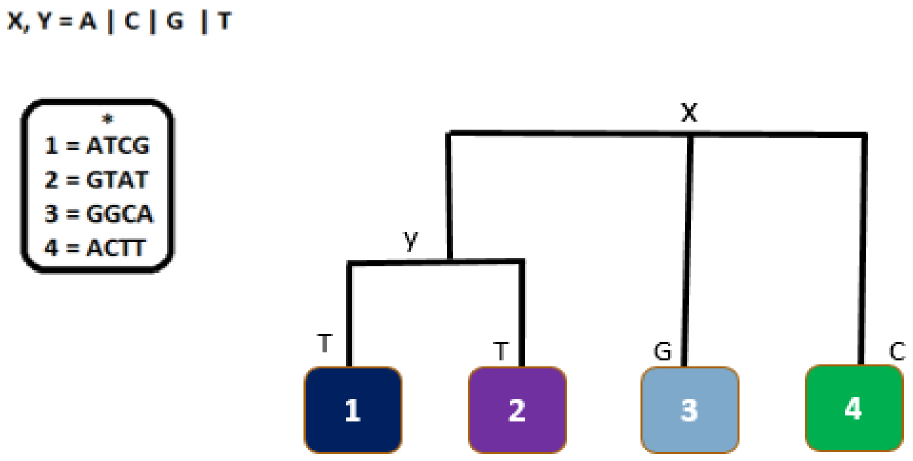

Fig. 3 Maximum Likelihood

For simplicity of explanation, Juck Cantor (JC) substitution model is used. In this model, the transition between the same bases are 0.7 and the different bases are 0.1. Prior probability for nucleotide ‘A’, ‘C’, ‘G’, ‘T’ at root node is 0.25 each. For given data equation 4 iterates 16 times for each site as given tree has two intermediate nodes. Therefore, total iteration is 64 times. Table 1 shows the likelihood calculation simulation for the 1st column of the sequence according to equation 3.

$$\log(P) = \sum \log(P_i) \qquad (6)$$

According to equation 6 log-likelihood for four column sequence is calculated and presented in Table 2. From Table 2, total log-likelihood of a given tree is -30.0821. Log-likelihood is used because values of likelihood probability are fractional.

Table 1: Simulation of likelihood calculation.

XY	P_{XY}	P_{XY} values
AA	P(A)×P(AA) ×P (AA) ×P (GA) × P(GA) × P(AA)	0.25 × 0.7×0.7×0.1 × 0.1 × 0.7 = 0.008575
AC	P(A) ×P(AC) × P (AC) ×P (GC) ×P(GA) × P(AA)	0.25 × 0.1 ×0.1 ×0.1×0.1 × 0.7 = 1.75e-05
AG	P(A) × P(AG) ×P (AG) × P (GG) × P(GA) × P(AA)	0.25 × 0.1 × 0.1 × 0.7 × 0.1 × 0.7 = 0.0001225
AT	P(A) ×P(AT) × P (AT) × P (GT) × P(GA) ×P(AA)	0.25 × 0.1 × 0.1 × 0.1 × 0.1 × 0.7 = 1.75e-05
CA	P© × P(CA) × P (AA) × P (GA) ×P(GC) × P(AC)	0.25 × 0.1 × 0.7 × 0.1 × 0.1 × 0.1= 1.75e-05
CC	P(C) × P(CC) ×P (AC) × P (GC) × P(GC) × P(AC)	0.25 ×0.7 × 0.1 ×0.1 × 0.1 × 0.1 = 1.75e-05
CG	P(C) × P(CG) × P (AG) × P (GG) × P(GC)×P(AC)	0.25 × 0.1 × 0.1 ×0.7 × 0.1 × 0.1 = 1.75e-05
CT	P(C) × P(CT) × P (AT) × P (GT) × P(GC) × P(AC)	0.25 × 0.1 × 0.1 × 0.1 × 0.1 × 0.1 = 2.5e-06
GA	P(G) × P(GA) × P (AA) × P (GA) × P(GG) × P(AG)	0.25 × 0.1 × 0.7 × 0.1 × 0.7 × 0.1 = 0.0001225
GC	P(G) × P(GC) × P (AC) × P (GC) × P(GG) × P(AG)	0.25 × 0.1 ×0.1 × 0.1 × 0.7 × 0.1 = 1.75e-05
GG	P(G) × P(GG) × P (AG) × P (GG) × P(GG) × P(AG)	0.25 × 0.7 × 0.1 × 0.7 × 0.7 × 0.1 = 0.0008575
GT	P(G) × P(GT) × P (AT) × P (GT) × P(GG) ×P(CG)	0.25 × 0.1 × 0.1 × 0.1 × 0.7 × 0.1 = 1.75e-05
TA	P(T) × P(TA) × P (AA) × P (GA) × P(GT) × P(AT)	0.25 × 0.1 × 0.7 × 0.1 × 0.1 × 0.1 = 1.75e-05
TC	P(T) × P(TC) × P (AC) × P (GC) × P(GT) × P(AT)	0.25 × 0.1 × 0.1 × 0.1 × 0.1 × 0.1 = 2.5e-06
TG	P(T) × P(TG) ×P (AG) × P (GG) × P(GT) × P(AT)	0.25 × 0.1 ×0.1 × 0.7 × 0.1 × 0.1 = 1.75e-05
TT	P(T) × P(TT) × P (AT) × P (GT) × P(GT) × P(AT)	0.25 × 0.7 × 0.1 × 0.1 × 0.1 ×0.1 = 1.75e-05

Table 2: Likelihood value for each site.

Site Likelihood	**Values**
$\log p_1$	-7.06149
$\log p_2$	-7.06149
$\log p_3$	-9.0074
$\log p_4$	-7.06149
Sum	-30.0821

Chapter III

Literature Review

3.1 Introduction

Phylogenetic tree construction problem leads to many probable partial solutions. They are limited to used data or sequences, time and resource. Some methods fit where species are greater than 40 but need huge computational cost [16]. Others are suited for a limited number of species such as below 40. Different mathematical, statistical and bio-inspired algorithms (ACO, GA, DPSO) are applied to get good results. More works on probabilistic and statistical methods are done. However, their performance leads researchers to look for bio-inspired algorithms. In recent years, there are few works done to solve phylogenetic tree reconstruction problem using ACO, GA, DPSO. We will briefly discuss existing works in this section.

3.2 Protein Phylogenetic Inference Using Maximum Likelihood With A Genetic Algorithm

Matsuda [17] proposed an approach for inferring phylogenies in which the author developed a genetic algorithm for maximum likelihood approach using EF-1α amino acid sequences. The alignment of these sequences are made by using “CLUSTAL W”. It is the first attempt to infer phylogenetic tree using genetic algorithm. In this process, branch exchanging of phylogeny trees used for the mutation operator and a crossover operator assigned based on genetic distances. This method gives probable alternative trees, high-resolution values for comparing the accuracy of generated trees. Author has compared the result with UPGMA, NJ and MP algorithm.

- **Advantages:** This algorithm is the first approach for phylogeny inference using a genetic algorithm. Though they are not close to global optimal, results are comparable to existing works.

- **Disadvantages:** But they have used a weak experimental dataset. Parameters of the genetic algorithm are not determined for the optimal solution. Therefore, parameter settings are user input.

3.3 A Genetic Algorithm for Maximum-Likelihood Phylogeny Inference Using Nucleotide Sequence Data

Lewis [18] has developed a genetic algorithm (GAML) based on another GA [17]. Lewis improved performance by limiting the use of branch-length optimization with maximum likelihood calculation. As individuals are sorted according to the fitness function, more offspring's of a good tree is added to the next population. For topological recombination, subtree pruning and regraft (SPR) is assigned. For the experiment, the RBCL_55 dataset is used.

- **Advantages:** GAML has performed same on rbcl_55 datasets with conventional heuristic search by taking only 6% computational resource. They have used the HKY model for maximum likelihood criterion. They had compared their algorithm with PAUP[7]. Three runs of the GA algorithm take 42.4h. To get the same result PAUP package take 783.2h (14 times as much computing effort).

- **Disadvantages:** For the initial population, the random tree is used. This can degrade the convergence of GAML. Only one dataset is used which have 55 species.

3.4 The Meta-Population Genetic Algorithm: An Efficient Solution for The Problem of Large Phylogeny Estimation

Lemmon and Millinkovich [15] have proposed a metapopulation genetic algorithm to estimate large phylogeny. In this process, the metapopulation genetic algorithm has used different populations of trees that are enforced to associate in the search for the optimal tree using inter-population consented information. Evaluation, selection and mutation events are subjected to a tree for each population. The inter-population communication is controlled by consensus pruning (CP). The author have used the HKY nucleotide substitution model as well as nested Jukes-Cantor (JC), K2P and F81 models to implement METAPIGA. This meta-population genetic approach is very accurate and faster than other heuristics approaches. They have used a maximum likelihood model to compare 100 of taxa in practical computational time. Simulated datasets of 20, 40, 80, 160, or 320 taxa are used.

- **Advantages:** MetaPIGA is flexible, fast on medium datasets. Consensus pruning and branch support vector are incorporated for reaching global optimal faster.

- **Disadvantages:** It needs a vast computational time for large dataset.

3.5 A Discrete Particle Swarm Optimization Algorithm for Phylogenetic Tree Reconstruction

HUI-YING and others [16] have proposed a discrete particle swarm optimization algorithm to construct a phylogeny tree. For inference, they have used parsimony metrics. For the experiment, they used 25-sequences, involving sequences of the chloroplast gene rbcL from a diversity of green plants. DPSO performs well where a number of sequence in population is less than 40.

- **Advantages:** This method needs less searching space and an efficient process for using a small population.

- **Disadvantages:** The used dataset is weak and this method is not suitable for more than 40 species.

3.6 A Novel Approach to Phylogenetic Tree Construction Using Stochastic Optimization and Clustering

Qin, Chen et al [14] have proposed a method which merges the clustering and ant colony optimization algorithm together to reconstruct the phylogenetic tree. All species are represented as objects/nodes in a digraph. Then ACO algorithm is used to get strongly connected components in the graph. These strongly connected components are local optimization. In local optimization, the distance based clustering method is used. Again, ACO is used to find the best-fitted tree from the strongly connected components (which is a global optimization). They have used two simulated datasets and vertebrate database (obtained from the NCBI genome database) which contains 832 mitochondrial proteins from 64 vertebrates. Results of this algorithm and GA is compared.

- **Advantages:** It shows that the proposed algorithm is easy to implement and efficient compared to GA.

- **Disadvantages:** Only small datasets are used for the experiment.

3.7 Genetic Algorithm Approaches for The Phylogenetic Analysis of Large Biological Sequence Datasets Under The Maximum Likelihood Criterion

Zwickl [19] has proposed a genetic algorithm for reconstructing phylogenetic tree called GARLI. GARLI is equipped to generate maximum likelihood tree for large sequence data of nucleotides, amino acids, and codon sequences. For optimality, GARLI optimizes a branch if certain preset criteria are met. At first, an initial population of this genetic algorithm is assigned either by user input or random topology. Next, branch length and the alpha parameter are optimized. Then, the generation of the algorithm starts. Mutation and crossover are occurred to produce new offspring. Mutation can be four types such as topological mutation, evolutionary model parameter mutation, branch-length parameter mutation and topological recombination. Finally, again branch lengths are optimized. They have compared the performance of GARLI algorithm with similar kind of algorithms such as RAxML and PHYML to construct the tree. For comparison, they have used 5 different datasets such as Rana, Angio, Rbcl, 1000ARB and Gutell3845.

- **Advantages :** Large sequence datasets are used and GARLI is efficient on large datasets.

- **Disadvantages:** For fitness function, only likelihood score is used.

3.8 Multi-Objective Evolutionary Algorithms and Phylogenetic Inference With Multiple Data Sets

Poladian, Jermiin et al [20] have proposed a multi-objective evolutionary algorithm for phylogenetic inference [39]. Authors have proposed a multi-objective process of phylogeny optimization problem first. Therefore, they have taken four species and two datasets for the experiment. Simple Juck and Cantor nucleotide substitution model is adopted. Constrained Pareto sets are used for MOO(multi-objective optimization) evaluation.

- **Advantages:** This proposal is the first introduction of multi-objective optimization to phylogenetic inference and shows future opportunity of MOO's in phylogeny inference.

- **Disadvantages:** Only four species are used for experimentation.

3.9 A Multi-Objective Evolutionary Approach for Phylogenetic Inference

Cancino, Delbem et al [21] have proposed a multi-objective genetic algorithm based on NSGA-II. Their main target for developing PhyloMOEA is to get a non-dominated set of trade-off solution between ML and MP criterion. For individual encoding, they have used the Graph Template Library (GTL) to represent a tree. For the initial population, they have used maximum parsimony, likelihood trees for the initial population, similar to [4,21]. For ML calculation, Felsenstein [22] and MP Fitch[23] algorithms are used. The final population of this algorithm is optimized by a non-decreasing Newton-Raphson method prescribed by [24]. Here, Crossover operator is implemented same as [25]. Mutation operator uses well-known topological modifications NNI, SPR, TBR, described in [23]. They have used rbcl_55, mtDNA_186, RDPII_218, ZILLA_500 datasets.

- **Advantages:** Alternative solutions for both criteria have been found on those datasets.

- **Disadvantages:** PhyloMOEA needs several hours to find acceptable Pareto-solutions; likelihood calculation ignores rate heterogeneity among sites and convergence metrics are not investigated.

3.10 A Multi-Criterion Evolutionary Approach Applied to Phylogenetic Reconstruction

Cancino, Delbem et al [5] proposed a Multi-criterion evolutionary algorithm applied to phylogenetic reconstruction problem. In this algorithm, authors have employed a genetic algorithm for finding the optimal solution. Two populations are used which is similar to the NSGA-II algorithm. For good convergence, they have provided initial solutions which are generated by maximum likelihood, maximum parsimony, and bootstrap analysis. Parsimony scores are calculated using the Fitch algorithm and Likelihood scores are calculated by Felsenstein algorithm. Multi-criteria fitness is evaluated by non-dominated sorting and crowding distance algorithm. Crossover and mutation operator evolves new populations from the parent population. For crossover subtree swapping as described in GAML[18] is implemented. NNI moves are implemented for mutation operation. For getting good likelihood value, a non-decreasing Newton-Raphson method is incorporated upon the final population to save execution time. For the experiment, they have used rbcl_55, mtDNA_186, RDPII_218, and ZILLA_500. To get good likelihood trees RAxML-V and PHYML program is used. For parsimony, NONA program is used. These likelihood and parsimony trees are provided in the initial population.

- **Advantages:** Convergence time is decreased by providing good bootstrapped initial tree.

- **Disadvantages**: Comparison between similar algorithm is not performed. Metrics for diversity and convergence has not been investigated.

3.11 Inference of A Phylogenetic Tree: Hierarchical Clustering Versus Genetic Algorithm

Blanchette and others [26] have compared performance and implementation of two computational methods agglomerative clustering and genetic algorithm. They have used Caminalcules artificial organisms for the experiment, which has 29 “currently existing species” and 48 “fossil species”. Their comparison shows that both methods perform almost same for Caminalcules species.

- **Advantages:** When problem size is larger GA takes much time.

- **Disadvantages:** Agglomerative clustering takes constant time, so for large dataset, it is a better choice.

3.12 Applying A Multiobjective Metaheuristic Inspired by Honey Bees to Phylogenetic Inference

Santander, Sergio, and Miguel *et al* [2] have designed a multiobjective metaheuristic algorithm inspired by honey bees (MOABC). Maximum likelihood and maximum parsimony criteria are used as objective functions. There is three type of bees in the proposed algorithm employee bees, onlooker bees, and scout bees.

- **Employee bee**: The initialization of the algorithm is assigning half of the swarm size bee’s random phylogenies from 1000 phylogenies (500 of them are created by maximum parsimony techniques and 500 of them are created by maximum likelihood). Now employee bees apply NNI moves to create new topology from their assigned topology. Then, using multi-objective fitness function the promising one from new and the previous one is kept.

- **Onlooker Bee:** Onlooker bees wait in dance area of the hive to get information from employee bees. Employ bees report best topology trees (solutions) to them. Then, they exploit the neighborhood of the topologies and keep the promising ones.
- **Scout Bee:** When topology of existing bees do not change for a while then scout bees search globally using PPN method.

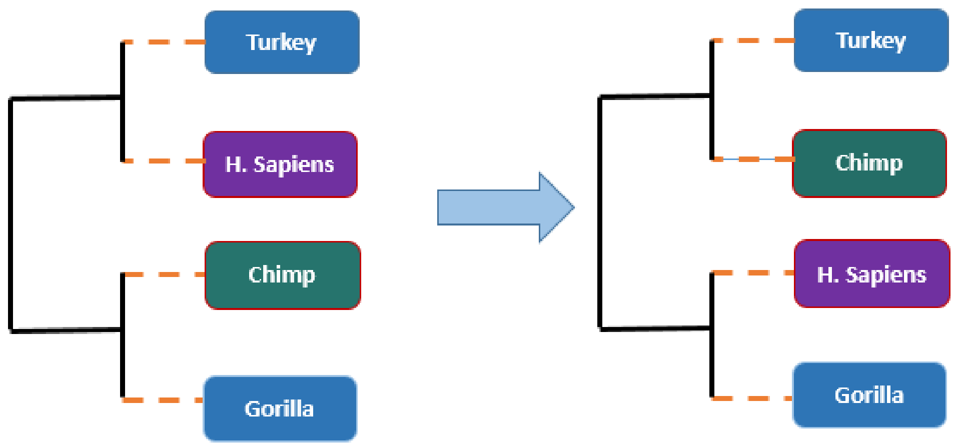

Fig. 4 NNI move.

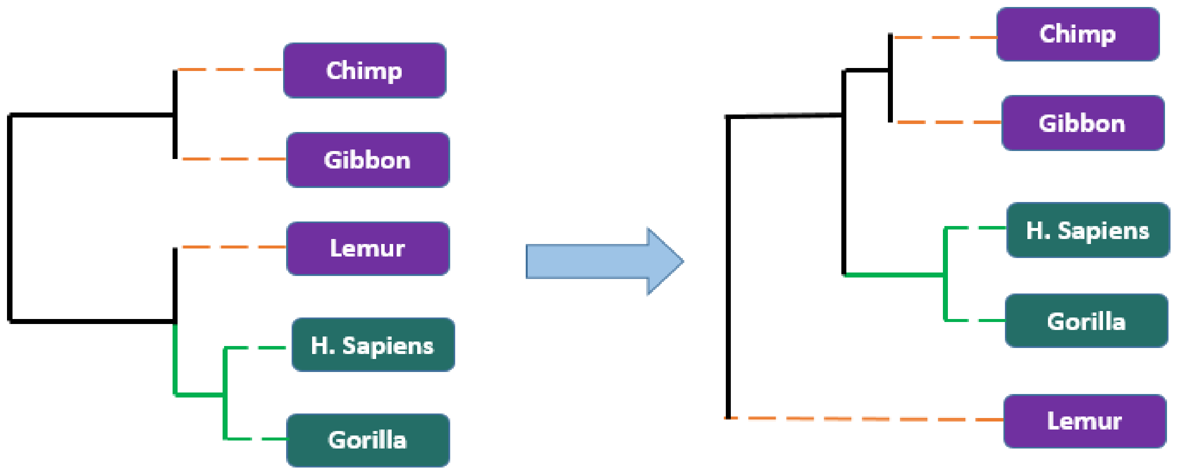

Fig. 5 SPR move.

Authors have used eight datasets to evaluate the performance of MOABC. Datasets are rbcL_55 (55 sequences, 1314 nucleotides per sequence of the rbcL gene), HIV2_72 (72 sequences, 828 nucleotides per sequence of HIV2), membracidae_81 (81 sequences, 3321 nucleotides per sequence of EF-1a and 28S rDNA), mtDNA_186 (186 sequences, 16608 nucleotides per sequence of human mitochondrial DNA), HIV1_192 (192 sequences, 817 nucleotides per sequence of HIV1), RDPII_218 (218 sequences, 4182 nucleotides per sequence of prokaryotic RNA), S1482 346 (346 sequences, 897 nucleotides per sequence of plant genus Tiquilia), ZILLA_500 (500 sequences, 759 nucleotides per sequence of rbcL plastid gene). These dataset is used to compute the likelihood score and parsimony score using RAxML, DNAML, GARLI, MetaPIGA, and MOABC. Comparison between likelihood scores and parsimony scores show that MOABC performs better in most datasets. To test the robustness of MOABC, Plethodontid salamander mitochondrial DNA dataset is used and MOABC is impressive in robustness. The drawback of this proposal is it gives a set of trees.

3.13 Well-Supported Phylogenies Using Largest Subsets of Core-Genes by Discrete Particle Swarm Optimization

Alssrraj and others [27] used chloroplast genomes to reconstruct the phylogenetic tree of the Rosales order. In order to reduce computational cost, a larger subset of the genome sequence is extracted to get most supported species tree. To extract sequence, they have used approaches for determining core-gene of chloroplast discussed in [28, 29]. For the experiment, Rosales order nine in-group species and one outgroup species are taken. In addition, DPSO metaheuristics are applied to find a Rosales order supported species tree with extracted 82 lengths of genes. In order to verify the result of the experiment same tree reconstruction is done using RAxML software and the topology supports the result of designed DPSO algorithm.

- **Advantages:** Authors have reduced computational cost by selecting a subset of the genome sequence.

- **Disadvantages:** Dataset and comparison of this proposal are poor. For experimental purpose chloroplast dataset and RAxML program are used for.

3.14 Using MOEA with Redistribution and Consensus Branches to Infer Phylogenies (MOEA-RC)

Min, Xiaoping et al have proposed a multi-objective evolutionary algorithm for redistribution and consensus branches to infer phylogenies in 2017[30]. In MOEA-RC, random trees or user given trees can be used as initial population. Then, MP and ML value of the population is calculated and updated. Next, the population is divided into sub-populations so that they can be sorted according to specific criteria. Consensus branches for each sub-population are discovered. Now, Prune-Delete-Graft is implemented as a crossover operator and Nearest-Neighbor-Interchange is used for mutation operator. The offspring is merged with the current population. This process continues until the stop condition is met. For experiment, rbcl_55, mtDNA_186 and ZILLA_500 is used. For multi-objective comparison PhyloMOEA, NSGA-II is used. Popular software packages MEGA 7(MP, ML), DNAPARS(MP) and Raxml(ML) are taken for comparison also.

- **Advantages:** MOEA-RC has performed better in terms of likelihood score.

- **Disadvantages**: It performs well only for one dataset(rbcl_55) considering parsimony score.

3.15 Evaluating Fast Maximum Likelihood-Based Phylogenetic Programs Using Empirical Phylogenomic Data Sets

Zhou, Shen et al [31] have compared four fast phylogenetic tree construction algorithms. They are PhyML, RAxML, FastTree, IQ-TREE. They are heuristics which try to explore problem space as minimum as possible with their use of NNI and SPR methods on feasible solution trees. In the earlier versions of PhyML, the only search is based upon NNI rearrangements only. However, later version of PhyML in 2010 has a mixture of SPR and NNI rearrangement. At first, SPR rearrangement with hill-climbing searching at earlier stages and then NNI rearrangement with hill-climbing searches is done in final stages. This later version improved the accuracy very much. RAxML implements SPR-based rearrangements with a hill-climbing algorithm for searching. It uses heuristics to optimize search space. RAxML has performed well in both accuracy and speed. RAxML and PhyML is very good in both speed and accuracy but more efforts are given to improve speed. As a result, another program named FastTree is developed. First, it makes approximate NJ tree which is then exploited with NNI and SPR rearrangements. At last, ML-based NNI rearrangements are done to get the final result. FastTree uses many heuristics to limit the run time of tree generation which is a trade-off between speed and accuracy. Another problem with above programs is that they have used a hill-climbing algorithm for searching which is very much susceptible to local optima. To solve this problem another program IQ-TREE have been developed. The latest version of IQ-TREE starts with multiple starting trees and chooses candidates from pool iteratively. Then, it performs random NNI moves to modify tree and initiate an NNI-based hillclimbing tree search. If a better tree is found, it is replaced otherwise iterated again. After unsuccessful iterations, the process terminates. The benefit of IQ-TREE is it uses lazy subtree rearrangement[3] and a highly optimized implementation of the likelihood function for better computational efficiency. According to this paper, IQ-TREE performed well than PhyML and RAxML where a number of taxa are fewer. Nevertheless, PhyML and RAxML givers better accuracy than IQ-TREE as they use SPR rearrangement mainly.

Chapter IV

Chemical Reaction Optimization for Phylogenetic Tree Construction

4.1 Introduction

Chemical reaction optimization is a population-based meta-heuristic algorithm developed by Albert Y.S. Lam [11]. It follows basic events of chemical reaction to optimize a given problem. There are mainly four operators in CRO. They are on-wall ineffective collision, decomposition, inter-molecular ineffective collision and synthesis. Molecules are basic elements of the CRO algorithm on which operations are performed to get the desired result. These molecules are actually solutions of an optimization problem. Each molecule has the molecular structure (ω), potential energy (PE), kinetic energy (KE) and number of hits. These attributes are very necessary for basic operations of CRO. There might be other attributes which will depend on implementations of elementary reactions. The molecular structure can be a variable, a vector, a matrix or any complex data structure to represent a solution. CRO operators are equipped with the power of simulated annealing and genetic algorithm operators. Therefore, it gets benefit from these algorithms. It can be modified to run in parallel. Potential Energy of a molecule is the objective function value of a molecule structure. If f() denotes the objective function value of molecule structure ω, then PE_{ω} = f (ω). Two ineffective collisions implement local search (intensification) and decomposition, synthesis implements diversification. This combination of intensification and diversification is perfect for finding the global optimum. CRO has two variables α and β for controlling ratio of intensification and diversification.

Four elementary type of reactions can take place to maintain energy conservation and redistribute energy among molecules. We have represented four elementary reactions in Figure 6.

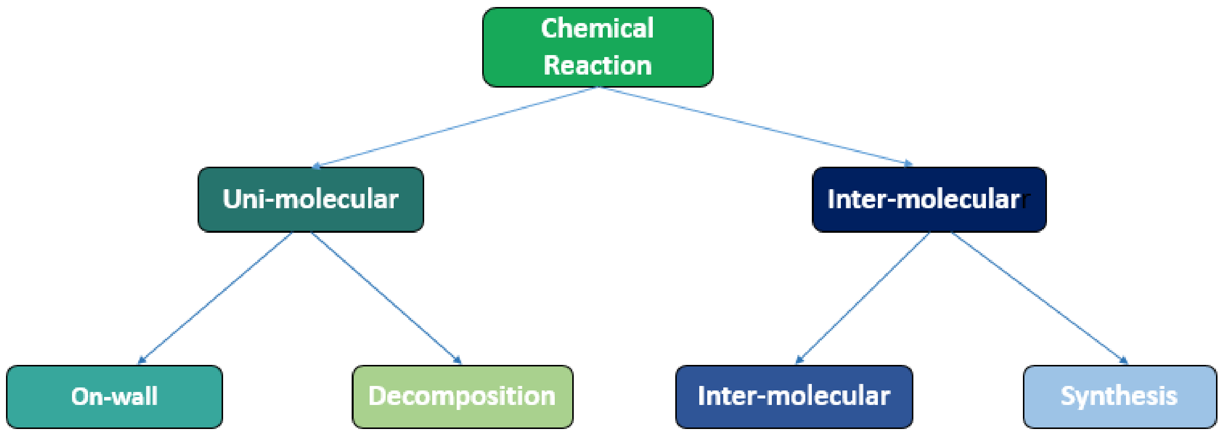

Fig. 6 Elementary reactions in CRO

On-wall ineffective collision: When one molecule collides with the wall of a container, then the internal structure of the molecule changes. Here, molecule m produces a new molecule m', i.e., m → m' [11]

Decomposition: In this elementary reaction, two new molecules are generated from a molecule. Two newly generated molecules bring diversity in their structure from the old molecule. Let, molecule m produces two new molecules m_1 and m_2, i.e., m → m_1 + m_2 [11].

Inter-molecular Ineffective collision: In an Inter-molecular ineffective collision, two molecules collide with each other. Let, two molecules m_1, m_2 collide with each other and produce two new molecules m_1' and m_2', i.e. m_1 + m_2 → m_1' + m_2'. This is much similar to On-wall ineffective collision except that the number of molecules is twice here [11].

Synthesis: Synthesis operator consolidates two molecules to form a new molecule. It is a reverse procedure of decomposition. Let m_1 and m_2 be two molecules. After the collision, m' molecule is created, i.e., m_1 + m_2 → m' [11].

4.2 CRO Strategy

We have already known that the strategy of CRO is totally dependent on the behavior during a chemical reaction. It has three primary stages throughout the execution of the whole algorithm. They are the initialization stage, iteration stage and the final stage. At the very beginning, which is the initialization stage, a number of initial parameters are selected. Some of the parameters are given in the table 3.

Table 3: Required parameters in CRO

Symbols	Algorithmic Definition
Pop Size	Population Size (Solution Space)
KELossRate	Kinetic Energy (KE) loss rate
Molecoll	Decide whether the chemical reaction is uni-molecular or inter-molecular
Buffer	Initial energy in the surroundings
InitialKE	Initial kinetic energy
α and β	Threshold values controlling the intensification and diversification
NumHit	Total number of hits a molecule has taken
Minstruct	Structure with minimum potential energy
MinPE	The potential energy when a molecule has minstruct
MinHit	The number of hits when a molecule has minstruct

4.3 Solution Representation

Structure of a molecule is composed of nodes. There are three types of nodes. They are a root node, hypothetical nodes, and leaf nodes. Nodes remember their pointers for their child nodes and parent node. Leaf nodes have no child and they represent species. Figure 7 illustrates a molecular structure/solution of our phylogenetic tree reconstruction problem. The tree has species A, B, C, and D. Nodes for this species are called leaf nodes. Leaf nodes have no right child or left child. Parents of A, B species and C, D species are called hypothetical nodes followed by a root node in the topmost**.**

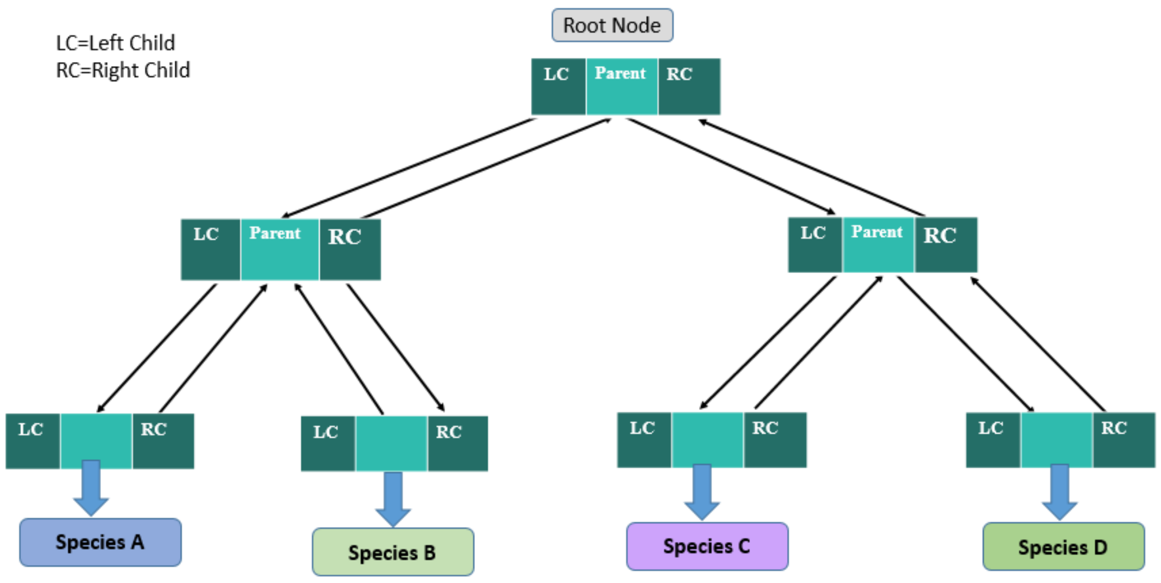

Fig. 7 Solution Representation**.**

4.4 Initialization and Population Generation

PT_CRO has two objective functions. It optimizes only one at a single run. Random phylogenetic trees are likely to local optimum and make optimization algorithm very ineffective. Therefore, literature[30] suggests to provide a set of good initial solution. As PT_CRO has two objective functions, 160 trees (half from PHYML[12] and half from BIONJ[43]) is provided as repository of solutions. Then, PT_CRO randomly selects 30 trees and initializes population. For maximum parsimony, PT_CRO performs its four reaction operators and one synthesis validation operator to find optimal solution. Similarly, for maximum likelihood, PT_CRO applies its four reaction operators and two repair operators.

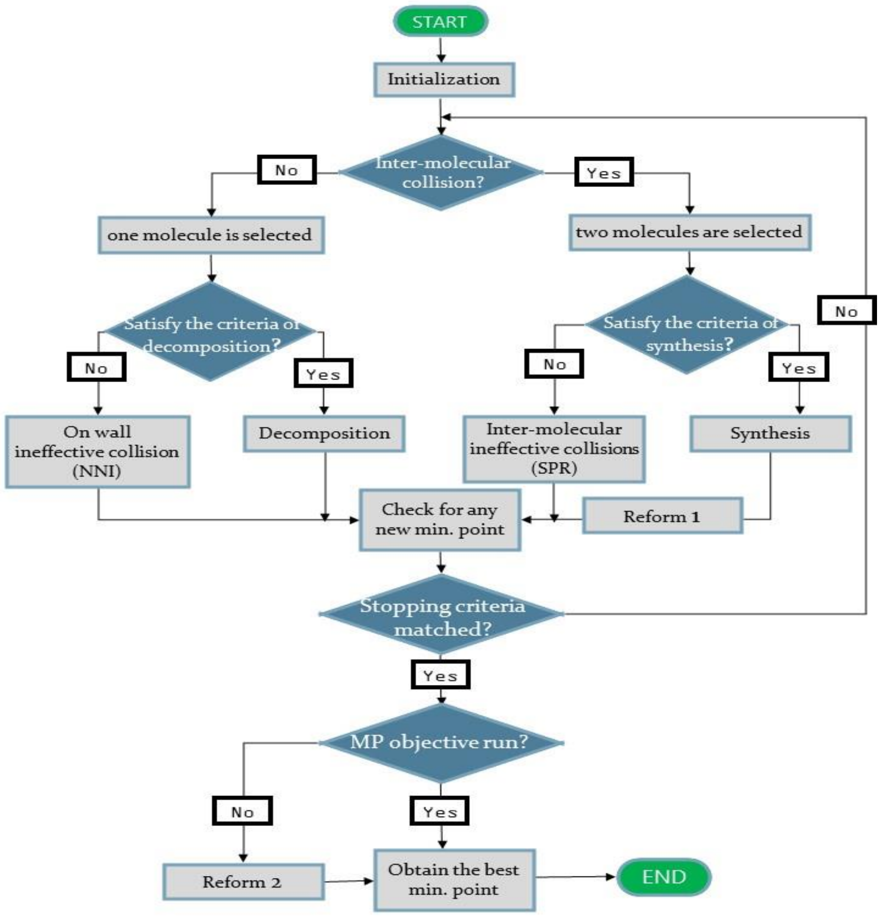
START
Initialization
Inter-molecular collision?
No
Yes
one molecule is selected
two molecules are selected
Satisfy the criteria of decomposition?
Satisfy the criteria of synthesis?
No
Yes
No
Yes
No
On wall ineffective collision (NNI)
Decomposition
Inter-molecular ineffective collisions (SPR)
Synthesis
Reform 1
Check for any new min. point
Stopping criteria matched?
Yes
MP objective run?
No
Yes
Reform 2
Obtain the best min. point
END

Fig. 8 CRO algorithm for Phylogenetic Tree Construction

4.5 Iteration and Operator design

In iteration step, molecules from initialization step will face the four elementary operations depicted in figure 8. These operators will explore the solution space of phylogenetic trees. In addition, the more relevant phylogenetic tree will be revealed. Our proposed PT_CRO algorithm has two differences from the basic CRO algorithm. The differences are that we have used reform1 operator on each synthesis solution and at the end of the iteration stage for ML run reform2 is applied further to improve likelihood values. The pseudo code of PT_CRO is shown in **Algorithm 1.**

Algorithm 1: PT_CRO

```
1. Input: Problem-Specific information (Objective Function, constraints, and initial population).
2. Compute the fitness value of each molecule as PE.
3.  While the stopping criteria not met do
4.      Generate t $\in$[0,1]
5.     if t > MoleColl  and pop_size < 2 then
6.             Randomly select one molecule m
7.           if decomposition criterion met then
8                 ($m_1, m_2$) = Decomposition(m)
9.           else
10.               $m'$ = On-wall Ineffective Collision (m)
11.          end if
12.     else
13.          Randomly select two molecules $m_1, m_2$
14.            If synthesis criterion met then
15.                      $m'$ = Synthesis($m_1, m_2$)
16.                      $m'$=Reform1($m'$)
17.          else
18.                ($m_1, m_2$) = Inter_molecular Ineffective Collision($m_1, m_2$)
19.         end if
20,    end if
21.   Check for any new minimum solution
22. end while
23. Reform2() on final population
24. Output: Overall minimum solution and its objective function value
```

4.5.1 On-wall Ineffective Collision

On-wall ineffective collision serves the purpose of local search in CRO. For our on-wall collision, we have adopted NNI (Nearest Neighbor Interchange) operator from [15]. At first, NNI selects two nodes whose grandparent is same and then interchanges their positions. Similarly, our on-wall operator first selects two nearest neighbor nodes of a tree whose grandparents are same. Then the positions of two selected nodes in the tree are swapped. Figure 9 visualizes on-wall ineffective collision. There we can see that solution m has a tree of four species P, Q, R and S. On-wall ineffective operator selects the nodes of Q and R species. Q and R have the same grandparent. So the positions of node Q and R are exchanged in solution m. In this way, new solution m' is formed. **Algorithm 2** presents the pseudo-code of the on-wall ineffective collision.

Algorithm 2: On-wall ineffective collision (m)

```
1. Input: Solution m
2. Output: $m'$
3. Copy m to form $m'$
4. While two random nodes have different grandparents
5. Select two nodes node1 and node2 randomly
6.    if node1,node2 have same grandparent then
7.        Exchange position of node1 and node2 in solution $m'$
8.         end while loop
9.    end if
10. end while
```

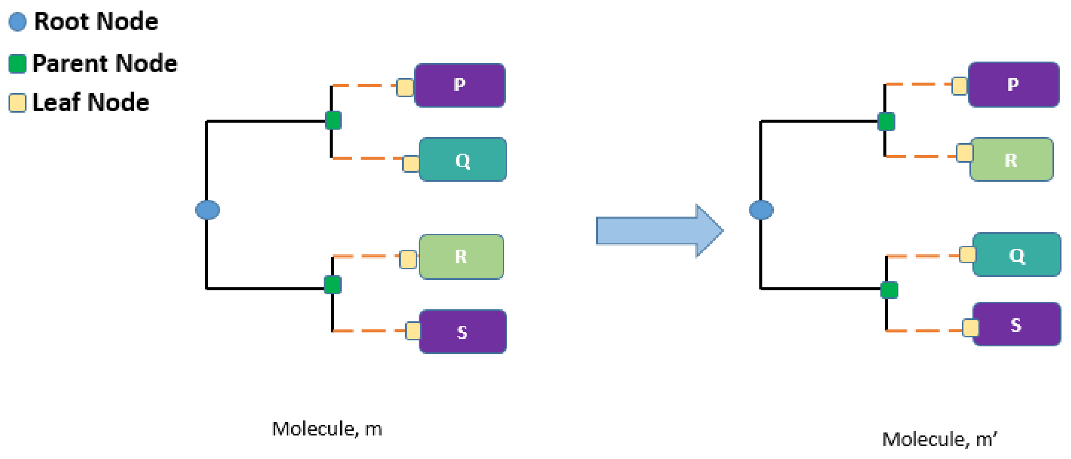

Fig. 9 On-wall Ineffective Collision

4.5.2 Decomposition

After exploring neighbors of the existing solution in the population extensively, Chemical Reaction Optimization (CRO) goes for creating and searching new solutions to skip local optima. Our decomposition operator takes a solution and creates two solutions, which contain significant subtree from parent solution. To explain this, let us assume solution m has a tree of seven species P, Q, R, S, X, Y, Z. Solution m contains two subtrees ((P, Q), (R, S)) and (X, (Y, Z)). In Figure 10, solution m_1 contains a subtree of solution m that are (X, (Y, Z)) and a random subtree generated from species P, Q, R, S. Then, a root node connects two subtrees obtained from solution m to form the solution m_1. Again, a subtree of solution m is copied ((P, Q), (R, S)) and a random subtree generated from species X, Y, Z. Next, a root node connects two subtrees obtained from solution m to form the solution m_2. Therefore, solution m is decomposed into two new solutions m_1 and m_2. **Algorithm 3** presents the pseudo-code of decomposition.

Algorithm 3: Decomposition (m)

```
1. Input: Solution m
2. Output: Solution $m_1$ and $m_2$
3. Select a random node containing 25% species of solution m
4. Copy the subtree of selected random node as u
```

5. Create a random subtree v using absent species at subtree u.

6. Create a root node at subtree u and v as child of the root node

7. Copy tree of the root node to solution m_1

8. Repeat step 3 to 7 for solution m_2 again.

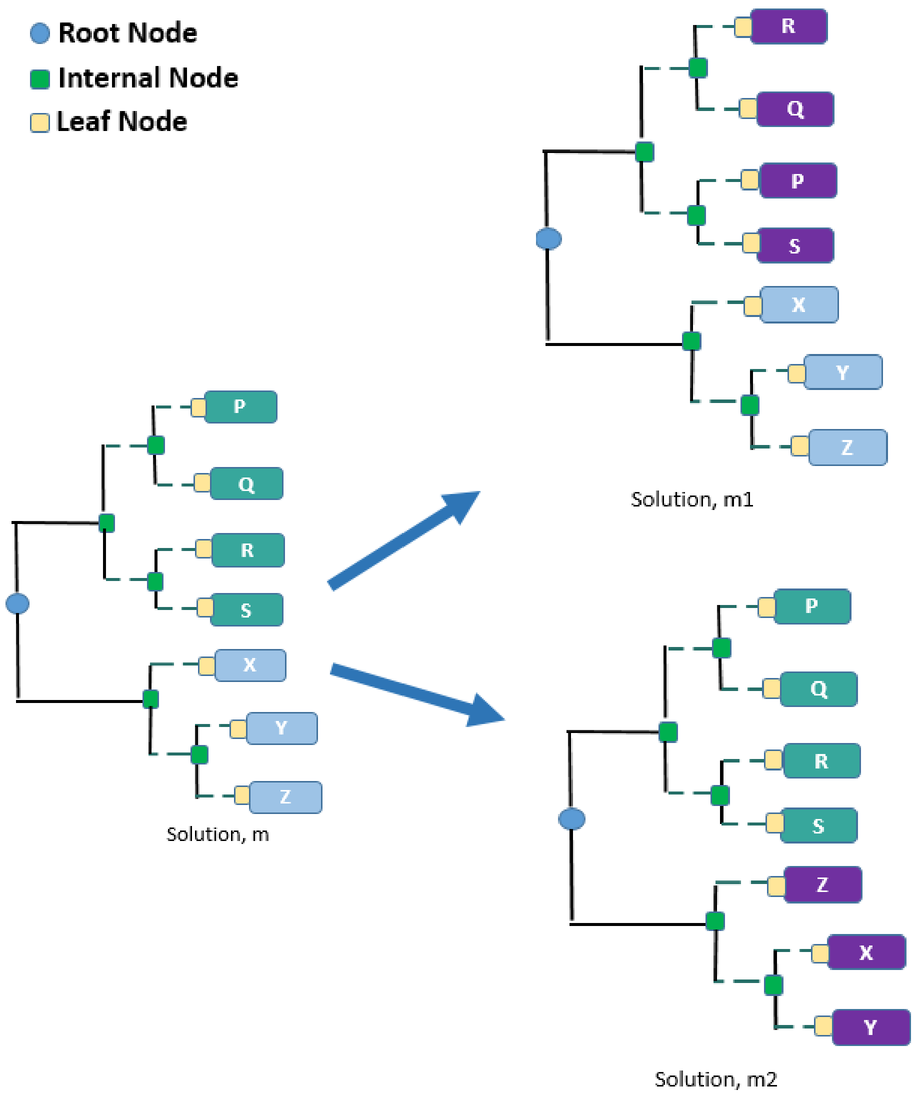

Fig. 10 Decomposition

4.5.3 Inter-molecular Ineffective Collision

This operator works on two solutions and modifies them to some extent. Therefore, the number of solutions in population remains the same in this operator. The inter-molecular collision is a local search operator. Our inter-molecular ineffective collision implements subtree pruning and regrafting (SPR) [19]. Subtree pruning and regrafting (SPR) is a technique to select a subtree and replace it to a new position in the tree. For implementation, two random nodes are selected one of which has grandparent and another node cannot be a leaf node. In Figure 11, the working principal of SPR move is demonstrated. Two nodes (node 7 and node 3) are selected randomly to perform subtree prune and re-graft. Node 7 has grandparent and node 3 is a leaf. Therefore, these two nodes fill the requirements of performing SPR. Sibling of node 7 is node 5 and parent of node 7 is node 4. Node 5 is replaced at the position of node 4. Node 4 is replaced at the position of node 3. Now node 3 is placed as sibling of node 7 and child of node 4. For example, solutions m_1 and m_2 are selected randomly. Then Subtree pruning and regrafting (SPR) technique discussed above is applied on solution m_1 and m_2. So two new solutions m_1' and m_2' are found. In Figure 12, solution m_1 and m_2 contain five species each. Initially, m_1 solution contains a tree ((P, Q), (R, (S, T))). After doing SPR, operation on m_1 the subtree (S, T) changes its position and becomes a new solution m_1'. Similarly, solution m_2 contains a tree (((P, Q), (S, T)), R). After doing SPR, operation on m_2 the subtree (S, T) changes its position and makes a new solution m_2'. Thus, the two new solutions m_1' and m_2' are generated. **Algorithm 4** presents the pseudo-code of the inter-molecular ineffective collision.

Algorithm 4: Inter-molecular ineffective collision (m_1, m_2)

```
1. Input: Two solutions $m_1$ and $m_2$
2. Output: $m_1'$ and $m_2'$
3. Copy solution $m_1$ into $m_1'$ and $m_2$ into $m_2'$
```

4. Select two nodes randomly from solution m_1', node 1 has grandparent and node 2 is not a leaf

5. Place sibling node of node 1 at their parent node's position

6. Place parent node of node 1 at the position of node 2

7. Add node 2 as the child of node 1's parent

8. Repeat step 3 to 7 for solution m_2'

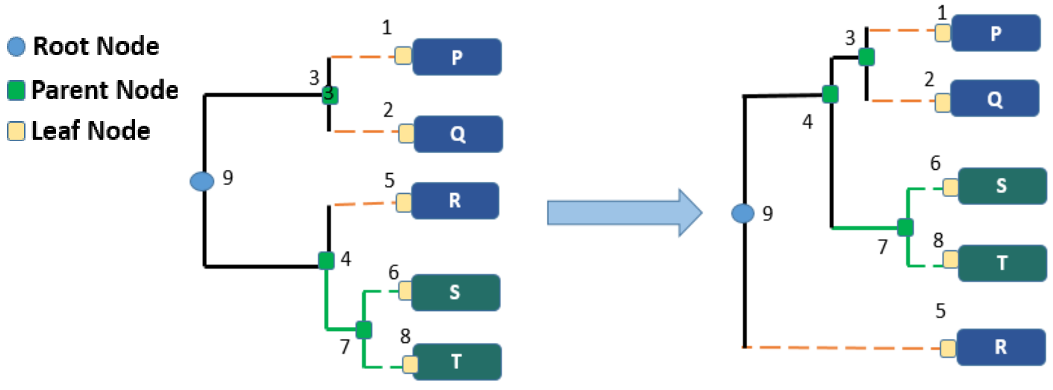

Fig. 11: SPR move

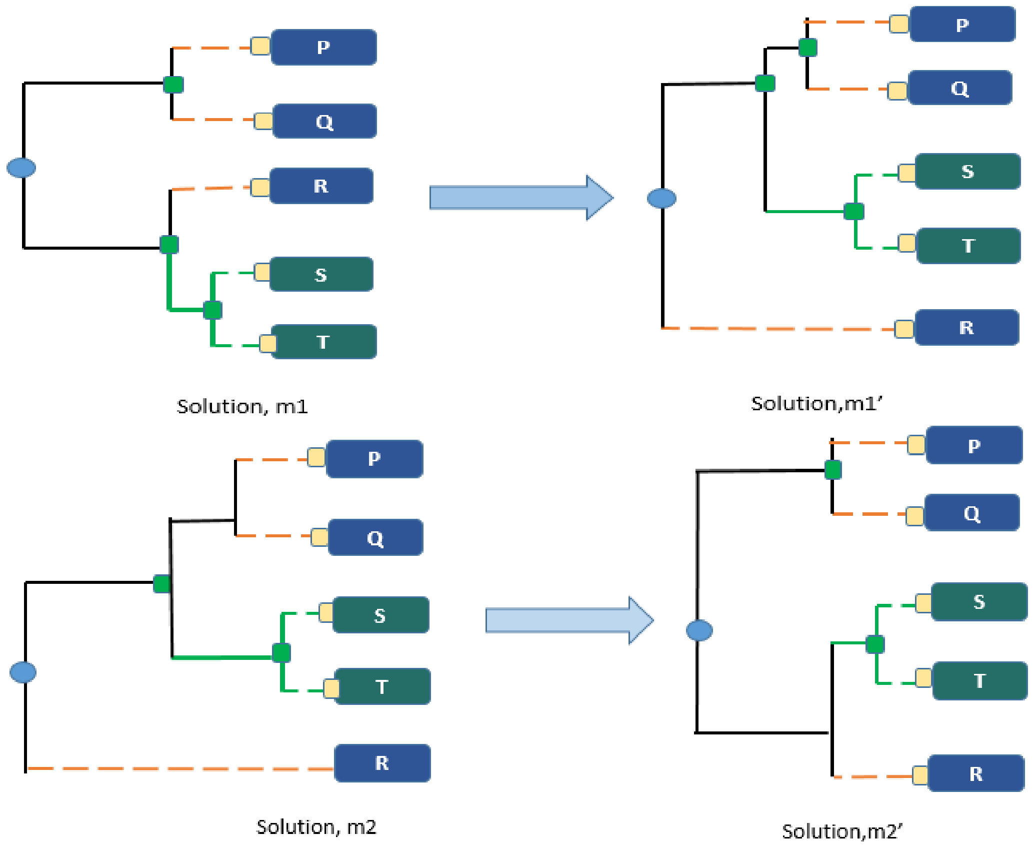

Fig. 12 Inter-molecular Ineffective Collision

4.5.4 Synthesis

When the synthesis reaction occurs, two molecules are used as input and they collide with each other and make a new molecule or solution. This is the reverse procedure of decomposition. Synthesis operates on two solutions and creates a new solution. Purpose of synthesis is to skip local trap. When existing solutions stall to improve for a while, they are sent to the synthesis operator. The operational approach for designing synthesis operator is a random selection of two solutions and making a new single solution. For this work, from two solutions two subtrees are merged together and formed a new solution. Figure 13 shows two solutions m_1 and m_2. Solution m_1 and m_2 have tree structure (((P, Q), (R, X)), (Y, (Z, S))) and (((R, Q), (P, S)), (X, (Y, Z))) respectively. The solution m_1 has two subtrees. They are b1 which has tree structure ((P, Q), (R, X)) and b2 which has tree structure (Y, (Z, S)). Moreover, m_2 has also two subtrees. They are b3 which has tree structure ((R, Q), (P, S)) and b4 which has tree structure (X, (Y, Z)). For example, we select subtree b1 and subtree b4 from solution m_1 and m_2 respectively. Two subtrees b1 and b4 are added as child of a root node and new solution m is formed. **Algorithm 5** presents the pseudo-code of synthesis.

Algorithm 5: Synthesis (m_1, m_2)

```
1. Input: Two solutions m1 and m2
2. Output: Solution m
3. Copy a random subtree u from solution m1
4. Copy a random subtree v from solution m2
5. Create a new root node
6. Add subtree u and v as a child of the root node
7. New solution m is found
8. Reform1(m)
```

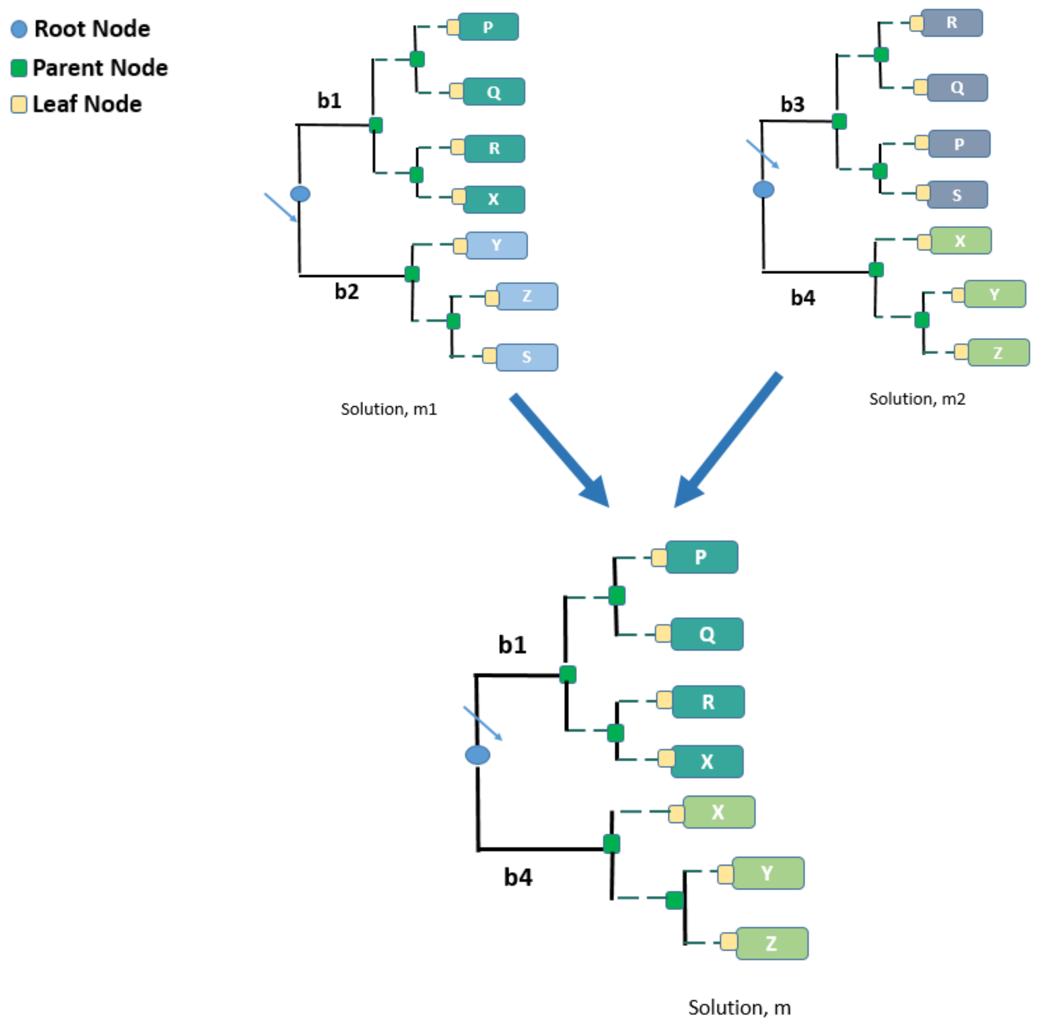

Fig. 13 Synthesis

4.5.5 Reform1

During synthesis operation, it is possible that duplicate species remain in solution and some species are absent. Therefore, the reform1 operator is designed so that duplicate species are deleted and absent species are added to make the solution valid. In Figure 14, solution m has duplicated species 'X' and species 'S' is absent. Therefore, Reform1 will delete species 'X' and add species 'S' to complete solution m.

As solution in synthesis, operation is created by randomly picking two subtree from two solution, so there is no surety of getting all species in new tree. In order to get valid solution, duplicate species should be removed and absent species should be inserted. Reform1 algorithm repairs final solution of synthesis operator to return valid solution tree.

Algorithm 6: Reform1 (m)

```
1. Input: One solution m
2. Output: m'
3. Find all duplicated species in solution m
4. Delete duplicated species
5. Find species, which is absent in solution m
6. Create a random subtree using absent species
7. Add random subtree to solution m
8. Copy solution m to new solution m'
```

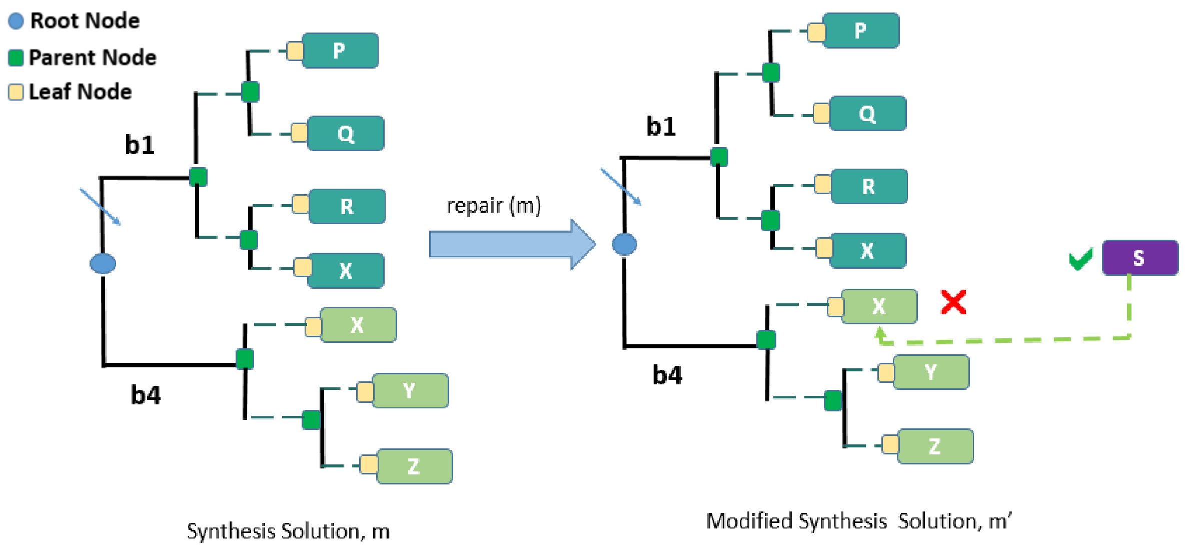

Fig. 14 Repeated Species Deletion and missing species insertion

4.5.6 Reform2

For improving log-likelihood value, branch length optimization using Newton-Raphson optimization algorithm is applied[31-33]. For better convergence, second derivative of the Newton-Raphson method is deployed. Newton-Raphson algorithm finds maximum or minimum of a function given function parameters. This method starts guessing at a random point. Then, iteratively changes parameter value to reach minimum. Newton-Raphson optimization for function f(x) follows the following equation 7

$$x_i = x_{i-1} - \frac{f'(x)}{f''(x)} \qquad (7)$$

x_i is the parameter of function f(x) which needs to be optimized. For branch length optimization using Newton-Raphson f(x) becomes the likelihood function and x_i is all branch lengths, one at a time. New optimized value is updated until tolerance level is reached. All branch lengths of phylogenetic tree are not optimized simultaneously because updating a branch length only changes partial likelihood of a tree. So, Likelihood function is optimized with only one branch at a time. First and second order derivative of likelihood function is used. For a bifurcating tree, the number of branches is 2n - 2. Therefore, a tree of five species will have eight branches. Now, the eight branches are parameters and likelihood function of equation 2, is the function to optimize. In the reform2 algorithm, a Tree faces Newton-Raphson optimization for branch lengths to improve likelihood. For further improvement, all possible nearest neighbors are interchanged. If partial likelihood value of a subtree improves, then new changed topology becomes the current tree. In Figure 15 a tree with five species and eight branches are shown. The reform function will optimize branch lengths to improve total likelihood. Next, a hill-climbing approach for searching all possible nearest node interchange is performed. At last, an improved likelihood tree is found.

Branch lengths of solution tree are unchanged during PT_CRO run. However, branch length has impact on maximum likelihood estimation. Therefore, Reform2 operator improves branch length values in order to improve ML. All possible Nearest-neighbor-interchange is performed to final solutions to improve likelihood value. But in PT_CRO only few nearest-neighbor-interchange is executed on solutions.

Algorithm 7: Reform2(m)

```
1. Input: One solution m
2. Output: m'
3. Apply Newton-Raphson optimization on solution m
4. Apply all possible nearest-neighbor-interchange for ML value improvement
5. Copy m to new solution m'
```

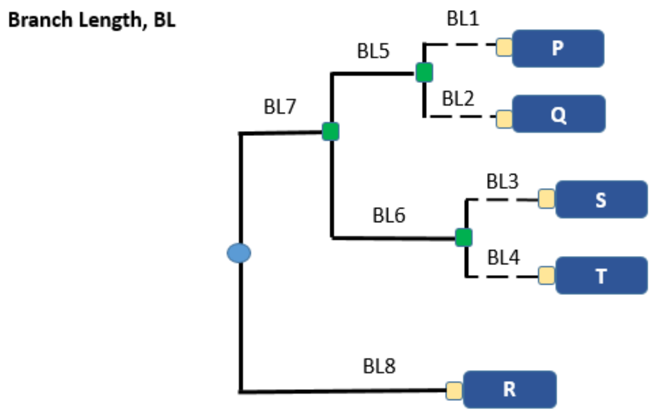

Fig. 15 A tree with Branch Length

4.5.7 Final stage

We will stop the iteration phase when a number of iterations exceeds a threshold value or objective value comes less than of a threshold value.

Chapter V

Experiment Results and Comparison with Existing Algorithms

5.1 Introduction

For our experiment, four datasets rbcl_55, mtDNA_186, RDPII_218 and ZILLA_500 are used. These datasets are obtained from MO-phylogeny website[34]. In Table 8, a number of sequences, sites per sequence are provided. Other metaheuristic algorithms have also used these four datasets. For maximum parsimony, PT_CRO is compared with MOABC, TNT, DNAPARS algorithm. In addition, for maximum likelihood version of PT_CRO MOABC, GARLI, METAPIGA, RAxML are used for comparison. For this experiment, HP Probook 450G1 personal pc is used with Intel® Core™i5-4200u CPU @ 2.50GHz (4CPUs), ~2.50GHz, 7.7GB RAM and running on Ubuntu 16.04.4 (64 bit). For the implementation, programming language was C++, compiler is gnu std c++11 and for text editing, Sublime Text had been used.

5.2 Experimental Setup

PT_CRO algorithm is implemented using a population size of 30 molecules, 100 as initial kinetic energy, maximum parsimony or maximum likelihood score as potential energy, 0.2 as molecular collision, 0.2 as kinetic energy loss rate. Controlling threshold for decomposition as 30 and controlling threshold for synthesis as 40 is set. Compared values of table 11 and 14 are obtained from MOABC. They have run MetaPIGA [4], Garli [19], DNAML [35] and RAxML [10] with following settings: population size 100, number of individual in population 25 and 100 generated trees. For TNT [36] and DNAPARS [6] program, sectorial search and tree fusing technique are used.

5.3 Parameter Settings used by CRO

For our proposed CRO algorithm to solve the PT_CRO phylogenetic tree construction problem, we at first initialized the parameters hit, alpha, beta, and kinetic energy KE. The values for these parameters are assumed initially as shown in table 4.

Table 4: Parameters used by PT_CRO in the experiment.

Parameters	Initial Values
Hit	0
Alpha (α)	40
Beta (β)	20
Kinetic Energy(KE)	100
KELossRate	0.2
MoleColl	0.2
Buffer	0
Iteration	1000
Pop_Size	30

5.4 Substitution Model and Rate Distribution

Purpose of substitution models for maximum likelihood is to provide realistic assumptions to get good likelihood scores. For our experiment, we have used General Time Reversible(GTR+Γ) model. For our four datasets, we have used different transition-transversion ratio which has been adopted from the website of MO-phylogenetics software[34]. The frequency of A, C, G and T and transition-transversion rates of each dataset are provided in Table 5.

Table 5: Transition-transversion values

Datasets	A	C	G	T	AC	AG	AT	CG	CT	GT
rbcl_55	0.310	0.158	0.166	0.363	1.723	5.126	0.661	1.954	8.368	1
mtDNA_186	0.310	0.316	0.129	0.243	1.584	58.555	1.181	1.661	39.614	1
RDPII_218	0.240	0.220	0.286	0.252	0.932	2.291	1.181	1.127	3.443	1
ZILLA_500	0.276	0.178	0.146	0.387	1.243	4.026	0.348	1.730	3.255	1

Variations of sites rate are modeled by assuming scaling of the branch length followed by gamma distribution. A number of categories, shape of parameters and rate for the individual database are presented in Table 6.

Table 6: Gama rate distribution parameters

Datasets	Number Of Categories	Shape (s)	Rate (r)
rbcl_55	4	0.3690	0.3690
mtDNA_186	4	0.05	0.05
RDPII_218	4	0.5450	0.5450
ZILLA_500	4	0.867	0.867

Table 7 expresses a sample structure of our four datasets. Here, each row represents a species and its 30 sites.

Table 7: Sample Dataset for rbcl_55

Species No.	Species	Sequences (Length-30)
1.	ANABAENA_S	CAAGATTACAGACTAACTTATTACACACCT
2.	CHARA_CONN	AAAGATTACAGATTAACTTACTATACTCCT
3.	SIROGONIUM	AAAGATTACAGACTTACATATTACACTCCT
4.	DUMORTIERA	AAAGATTATCGATTAACTTATTACACTCCG
5.	MARCHANTIA	AAAGATTATCGATTAACTTATTACACTCCG
6.	BAZZANIA_J	AAAGATTATAGATTAACCTATTATACGCCT
7.	PORELLA_4	AAAGATTATAGATCAACTTATTATACTCCC
8.	LEUCODON_5	AAAGATTACAGATTAACTTATTACACTCCA
9.	GINKGOBIL	AAAGATTACAGATTGACTTATTATACTCCT
10.	OSMUNDA_CI	AAAGATTATCGATTGACTCACTATACTCCC

In Figure 16 a tree has been drawn using the sample dataset of Table 7. Online newick to tree draw algorithm is used to draw this tree. Using the tree of Figure 16 maximum parsimony and maximum likelihood values for each site are calculated and presented in Table 8 and Table 9.

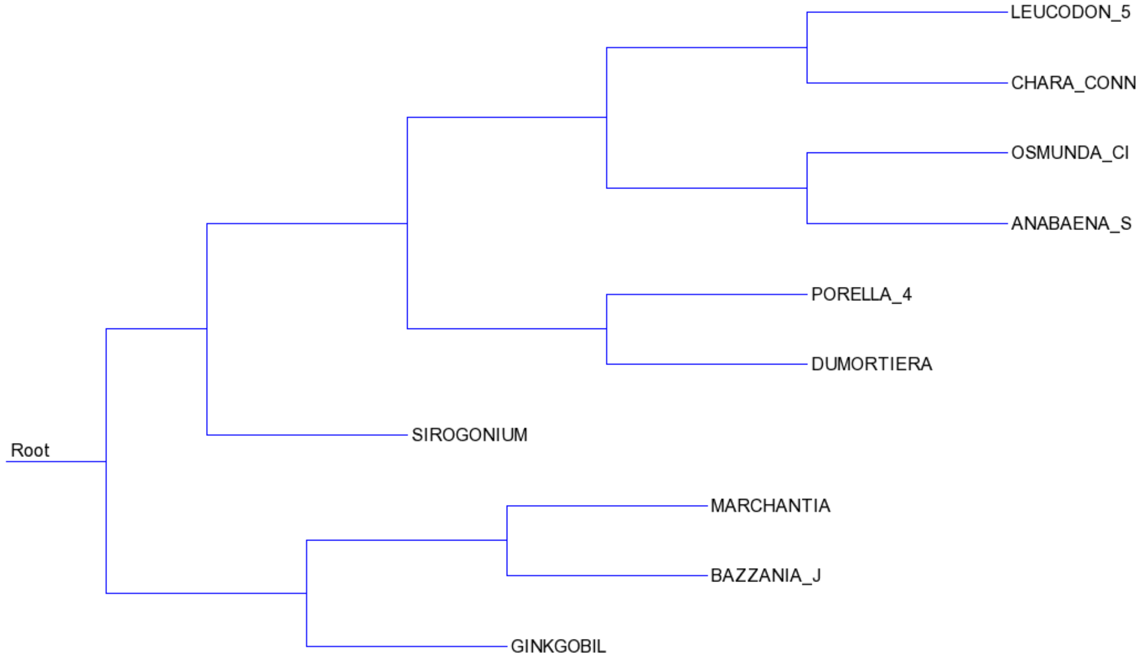

Fig. 16 A random tree for sample dataset in Table 7

Parsimony score calculation for each individual site of the tree in Figure 16 is given in Table 8. First column of Table 8 represents which site position is used. Second column shows sites for a similar position in sequences. Parsimony scores for each site and total parsimony value according to equation 2 for the tree of Figure 16 are presented in third column of Table 8

Table 8: Parsimony score for each column site

Column No.	Column Sites	Parsimony score for each column
1.	C,A,A,A,A,A,A,A,A,A	1
2.	A,A,A,A,A,A,A,A,A,A	0
3.	A,A,A,A,A,A,A,A,A,A	0
4.	G,G,G,G,G,G,G,G,G,G	0
5.	A,A,A,A,A,A,A,A,A,A	0
6.	T,T,T,T,T,T,T,T,T,T	0
7.	T,T,T,T,T,T,T,T,T,T	0

8.	A,A,A,A,A,A,A,A,A,A	0
9.	C,C,C,T,T,T,T,C,C,T	3
10.	A,A,A,C,C,A,A,A,A,C	3
11.	G,G,G,G,G,G,G,G,G,G	0
12.	A,A,A,A,A,A,A,A,A,A	0
13.	C,T,C,T,T,T,T,T,T,T	2
14.	T,T,T,T,T,T,C,T,T,T	1
15.	A,A,T,A,A,A,A,A,G,G	3
16.	A,A,A,A,A,A,A,A,A,A	0
17.	C,C,C,C,C,C,C,C,C,C	0
18.	T,T,A,T,T,C,T,T,T,T	2
19.	T,T,T,T,T,T,T,T,T,C	1
20.	A,A,A,A,A,A,A,A,A,A	0
21.	T,C,T,T,T,T,T,T,T,C	2
22.	T,T,T,T,T,T,T,T,T,T	0
23.	A,A,A,A,A,A,A,A,A,A	0
24.	C,T,C,C,C,T,T,C,T,T	5
25.	A,A,A,A,A,A,A,A,A,A	0
26.	C,C,C,C,C,C,C,C,C,C	0
27.	A,T,T,T,T,G,T,T,T,T	2
28.	C,C,C,C,C,C,C,C,C,C	0
29.	C,C,C,C,C,C,C,C,C,C	0
30.	T,T,T,G,G,T,C,A,T,C	5
		Total MP Score=30

Maximum likelihood value for each site of tree from Figure 16 is shown in Table 9 according to equation 3 and 4. Sites used for each ML calculation are also shown.

Table 9: Likelihood score for each column site

Column No.	Column Sites	Site Likelihood (P_i)	Log Likelihood (log P_i)
1.	C,A,A,A,A,A,A,A,A,A	0.000506412	-7.58816
2.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
3.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
4.	G,G,G,G,G,G,G,G,G,G	0.173247	-1.75304
5.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
6.	T,T,T,T,T,T,T,T,T,T	0.141139	-1.95801
7.	T,T,T,T,T,T,T,T,T,T	0.141139	-1.95801
8.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
9.	C,C,C,T,T,T,T,C,C,T	8.00626e-06	-11.7353
10.	A,A,A,C,C,A,A,A,A,C	1.25971e-07	-15.8872
11.	G,G,G,G,G,G,G,G,G,G	0.173247	-1.75304
12.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
13.	C,T,C,T,T,T,T,T,T,T	7.34462e-05	-9.51896
14.	T,T,T,T,T,T,C,T,T,T	0.00169672	-6.37906
15.	A,A,T,A,A,A,A,A,G,G	2.06538e-06	-13.0902
16.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
17.	C,C,C,C,C,C,C,C,C,C	0.120281	-2.11793
18.	T,T,A,T,T,C,T,T,T,T	2.66813e-05	-10.5315
19.	T,T,T,T,T,T,T,T,T,C	0.00169628	-6.37932
20.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
21.	T,C,T,T,T,T,T,T,T,C	7.20179e-05	-9.5386
22.	T,T,T,T,T,T,T,T,T,T	0.141139	-1.95801
23.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
24.	C,T,C,C,C,T,T,C,T,T	2.06549e-07	-15.3927
25.	A,A,A,A,A,A,A,A,A,A	0.140631	-1.96162
26.	C,C,C,C,C,C,C,C,C,C	0.120281	-2.11793
27.	A,T,T,T,T,G,T,T,T,T	9.8363e-06	-11.5294
28.	C,C,C,C,C,C,C,C,C,C	0.120281	-2.11793
29.	C,C,C,C,C,C,C,C,C,C	0.120281	-2.11793
30.	T,T,T,G,G,T,C,A,T,C	5.90739e-09	-18.9471
		Total P_i = 0.000506412	Total log P_i =7.58816

The four datasets that are used in experiment are mtDNA_186, RDPII_218, ZILLA_500, rbcl_55**.** These datasets are available in Table 10.

Table 10: Datasets used for the experiment

Dataset	Links	Total Sites	References
mtDNA_186	http://khaos.uma.es/mophylogenetics/datasets.jsp	$186 \times 16{,}608$	[2]
RDPII_218	http://khaos.uma.es/mophylogenetics/datasets.jsp	218×4182	[2]
ZILLA_500	http://khaos.uma.es/mophylogenetics/datasets.jsp	500×759	[2]
rbcL_55	http://khaos.uma.es/mophylogenetics/datasets.jsp	55×1314	[2]

5.5 Experimental Result and Comparison with the existing algorithm

5.5.1 Maximum Parsimony (MP)

For maximum parsimony, PT_CRO runs five times. Worst, average and best (on best result) of MP results are measured. Table 11 represents the measures.

Table 11: Statistical results for five PT_CRO run for MP score

Datasets	Worst	Average	Best
rbcl_55	4687	4681.2	4675
mtDNA_186	2178	2167.6	2158
RDPII_218	34379	34333.4	34278
ZILLA_500	16451	16396.6	16218

For maximum parsimony criterion, the best MP values after PT_CRO with reform2 of Table 12 are taken for comparison. PT_CRO has found a good score than MOABC [2], TNT [36], DNAPARS [6] in three datasets. On the other hand, for ZILLA_500 MP score of PT_CRO is equal to the good score. Table 12 shows the overall comparison of maximum parsimony with other algorithms.

Table 12: Comparison between existing algorithm and our algorithm for Maximum Parsimony (MP) values

DATASETS	**MOABC[2]**	**TNT[36]**	**DNAPARS[6]**	**PT_CRO**
rbcl_55	4874	4874	4874	**4675**
mtDNA_186	2431	2431	2431	**2158**
RDPII_218	41488	41488	41587	**34278**
ZILLA_500	**16218**	**16218**	16224	**16218**

PT_CRO for maximum parsimony run time is compared with TNT[36] program. Table 13 shows the time comparison and corresponding maximum parsimony value. These two programs are run on Ubuntu 16.04 (64 bit) operating system and same configuration personal computer as described in section 5.1.

Table 13: Time comparison among TNT and PT_CRO

Datasets	**TNT**		**PT_CRO**	
	Execution Time (in seconds)	**Result**	**Execution Time (in seconds)**	**Result**
rbcl_55	11	4874	3060	4675
mtDNA_186	23	2431	7200	2158
RDPII_218	804	41488	11736	34278
ZILLA_500	148	16218	8568	16218

5.5.2 Maximum Likelihood (ML)

For maximum likelihood, PT_CRO also runs five times. Worst, average, and best (on best results) of ML results are measured. In Table 14, we have shown the ML values using Reform2 and without using reform2 operator.

Table 14: ML values with and without Reform2 operator

Datasets	**ML values without using Reform2 operator**			**ML values using Reform2 operator**		
	Worst	**Average**	**Best**	**Worst**	**Average**	**Best**
rbcl_55	-22248.8	-22195.654	-22075.3	-21808.8	-21791.2	-21768.6
mtDNA_186	-40260	-40141.484	-39871.6	-39999	-39912.54	-39865.5
RDPII_218	-146689	-143980.6	-146226	-134570	-134526.2	-134475
ZILLA_500	-81688.3	-81416.9	-80633.2	-80848.3	-80695.31	-80592.40

Likelihood values of the four datasets with popular likelihood algorithms are compared in Table 15. Likelihood probabilities are fractional. Therefore, it is difficult to represent. Log-likelihood values of these fractional numbers are used as maximum likelihood score. The best ML values after PT_CRO with reform2 of Table 13 are taken for comparison. We have achieved a good score in contrast with other algorithms for rbcl_55 and mtDN_218 datasets. However, the result for the other two datasets is not up to the satisfactory level.

Table 15: Comparison between existing algorithm and our algorithm for Maximum Likelihood (ML) values

Datasets	**MOABC [2]**	**RAxML [10]**	**DNAML [35]**	**GARLI [19]**	**METAPIGA [4]**	**PT_CRO**
rbcl_55	-21,791.17	-21,788.56	-21,819.21	-21,769.12	-21,797.54	**-21,768.6**
mtDNA_186	-39868.94	-39,869.53	-39,891.78	-39,914.47	-41,939.54	**-39,865.5**
RDPII_218	**-1,34,073.89**	-1,34,077.70	-1,34,196.48	-1,34,077.50	-1,39,096.91	-134475
ZILLA_500	**-80585.04**	-80,590.00	-80,615.83	-80,606.09	-1,04,660.98	-80,592.40

Chapter VI

Conclusion and Future Directions

6.1 Conclusion

The phylogenetic tree construction problem is one of the standard problems in molecular evolution biology. This NP-hard problem has been widely used in biological as well as educational activities that draws attention of the researchers to solve this problem. Researchers proposed many optimal and near-optimal approaches but optimal approaches take exponential time. The near-optimal approaches using the approximation, heuristic, metaheuristic algorithms have been proposed to solve this problem. In this thesis, we have implemented an algorithm to solve phylogenetic tree construction problem using the concept of a population based metaheuristic chemical reaction optimization algorithm. Because of intensification and diversification characteristics of CRO, we have chosen it. Besides this, in recent times the success of CRO in different optimization problems motivates us to use it. We have implemented elementary operators of CRO and used two repair methods to solve this problem. The main contribution of our work is the designing of decomposition, synthesis operator and use of repair methods, which are applied on final solutions to get better results. For one thousand iterations we have tuned parameters for individual dataset. The results of PT_CRO have been compared with other metaheuristic algorithms for both maximum parsimony and maximum likelihood criteria such as artificial bee colony, genetic algorithm and metapopulation genetic algorithm. The results of PT_CRO are better in maximum parsimony for three datasets and same for one dataset. Also, in maximum likelihood for two datasets PT_CRO gets better result.

6.2 Limitations

From the experimental results, we can see that for maximum likelihood criteria ML (Maximum Likelihood) score is not better for two large datasets. The redesigning of reaction operators are not sufficient to maximize the ML scores. Therefore, the redesigning of reaction operators need to be revised. Besides, reform operators consume more execution time to calculate the ML score.

6.3 Future Direction

PT_CRO is designed to run on a single objective function at a time. However, multi-objective version can produce more significant tree topology alongside with improved parsimony and likelihood scores. However, calculating parsimony and likelihood are very memory and resource consuming. In single objective version, we either calculate MP or ML objective function value of solutions in a single run. Nevertheless, for multi-objective, we ought to calculate both MP and ML value simultaneously which is very time consuming. To resolve these constraints and form a resilient multi-objective algorithm that converges fast, parallel processing and PLL (Phylogenetic Likelihood Library) library should be incorporated.

Bibliography

1. Talib, Y., et al., *PHYLOGENETIC TREE CONSTRUCTION OF BIOSURFACTANT PRODUCING ORGANISMS.* Journal of Global Biosciences, 2016. **5**(5): p. 4105-4108.
2. Santander-Jiménez, S. and M.A. Vega-Rodríguez, *Applying a multiobjective metaheuristic inspired by honey bees to phylogenetic inference.* Biosystems, 2013. **114**(1): p. 39-55.
3. Stamatakis, A., *Phylogenetics: Applications, software and challenges.* Cancer Genomics-Proteomics, 2005. **2**(5): p. 301-305.
4. Helaers, R. and M.C. Milinkovitch, *MetaPIGA v2. 0: maximum likelihood large phylogeny estimation using the metapopulation genetic algorithm and other stochastic heuristics.* BMC bioinformatics, 2010. **11**(1): p. 379.
5. Cancino, W. and A. Delbem, *A multi-criterion evolutionary approach applied to phylogenetic reconstruction*, in *New Achievements in Evolutionary Computation*. 2010, InTech.
6. Felsenstein, J., *PHYLIP (phylogeny inference package), version 3.5 c*. 1993: Joseph Felsenstein.
7. Swofford, D.L., *Phylogenetic analysis using parsimony.* Illinois Natural History Survey, Champaign, Illinois, 1985.
8. Olsen, G.J., et al., *fastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood.* Bioinformatics, 1994. **10**(1): p. 41-48.
9. Guindon, S., et al., *Estimating maximum likelihood phylogenies with PhyML*, in *Bioinformatics for DNA sequence analysis*. 2009, Springer. p. 113-137.
10. Stamatakis, A., *RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies.* Bioinformatics, 2014. **30**(9): p. 1312-1313.
11. Lam, A.Y. and V.O. Li, *Chemical reaction optimization: A tutorial.* Memetic Computing, 2012. **4**(1): p. 3-17.
12. Snell, Q., et al. *Parallel phylogenetic inference*. in *Proceedings of the 2000 ACM/IEEE conference on Supercomputing*. 2000. IEEE Computer Society.
13. Goeffon, A., J.-M. Richer, and J.-K. Hao, *Progressive tree neighborhood applied to the maximum parsimony problem.* IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2008. **5**(1): p. 136-145.

14. Qin, L., et al., *A novel approach to phylogenetic tree construction using stochastic optimization and clustering.* BMC bioinformatics, 2006. **7**(4): p. S24.

15. Lemmon, A.R. and M.C. Milinkovitch, *The metapopulation genetic algorithm: An efficient solution for the problem of large phylogeny estimation.* Proceedings of the National Academy of Sciences, 2002. **99**(16): p. 10516-10521.

16. Lv, H.-Y., W.-G. Zhou, and C.-G. Zhou. *A discrete particle swarm optimization algorithm for phylogenetic tree reconstruction*. in *Machine Learning and Cybernetics, 2004. Proceedings of 2004 International Conference on*. 2004. IEEE.

17. Matsuda, H. *Protein phylogenetic inference using maximum likelihood with a genetic algorithm*. in *Pacific symposium on biocomputing*. 1996. World Scientific.

18. Lewis, P.O., *A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data.* Molecular biology and evolution, 1998. **15**(3): p. 277-283.

19. Zwickl, D.J., *Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion*. 2006.

20. Poladian, L. and L.S. Jermiin, *Multi-objective evolutionary algorithms and phylogenetic inference with multiple data sets.* Soft Computing, 2006. **10**(4): p. 359-368.

21. Cancino, W. and A.C. Delbem. *A multi-objective evolutionary approach for phylogenetic inference*. in *International Conference on Evolutionary Multi-Criterion Optimization*. 2007. Springer.

22. Felsenstein, J., *Evolutionary trees from DNA sequences: a maximum likelihood approach.* Journal of molecular evolution, 1981. **17**(6): p. 368-376.

23. Fitch, W.M., *Toward defining the course of evolution: minimum change for a specific tree topology.* Systematic Biology, 1971. **20**(4): p. 406-416.

24. Yang, Z., *Maximum-likelihood estimation of phylogeny from DNA sequences when substitution rates differ over sites.* Molecular biology and evolution, 1993. **10**(6): p. 1396-1401.

25. Felsenstein, J. and J. Felenstein, *Inferring phylogenies*. Vol. 2. 2004: Sinauer associates Sunderland, MA.

26. Blanchette, G., R. O'Keefe, and L. Benuskova. *Inference of a phylogenetic tree: hierarchical clustering versus genetic algorithm*. in *Australasian Joint Conference on Artificial Intelligence*. 2012. Springer.

27. Alsrraj, R., et al., *Well-supported phylogenies using largest subsets of core-genes by discrete particle swarm optimization.* arXiv preprint arXiv:1706.08514, 2017.

28. Alkindy, B., et al., *Finding the core-genes of chloroplasts.* arXiv preprint arXiv:1409.6369, 2014.

29. AlKindy, B., et al. *Gene similarity-based approaches for determining core-genes of chloroplasts*. in *Bioinformatics and Biomedicine (BIBM), 2014 IEEE International Conference on*. 2014. IEEE.

30. Min, X., et al., *Using MOEA with Redistribution and Consensus Branches to Infer Phylogenies.* International journal of molecular sciences, 2017. **19**(1): p. 62.

31. Zhou, X., et al., *Evaluating fast maximum likelihood-based phylogenetic programs using empirical phylogenomic data sets.* Molecular biology and evolution, 2017. **35**(2): p. 486-503.

32. Adachi, J. and M. Hasegawa, *MOLPHY version 2.3: programs for molecular phylogenetics based on maximum likelihood*. 1996: Institute of Statistical Mathematics Tokyo.

33. Bryant, D., N. Galtier, and M.-A. Poursat, *Likelihood calculation in molecular phylogenetics.* Mathematics of evolution and phylogeny, 2005: p. 33-62.

34. Zambrano-Vega, C., A.J. Nebro, and J.F. Aldana-Montes, *MO-Phylogenetics: a phylogenetic inference software tool with multi-objective evolutionary metaheuristics.* Methods in Ecology and Evolution, 2016. **7**(7): p. 800-805.

35. Hasegawa, M., H. Kishino, and N. Saitou, *On the maximum likelihood method in molecular phylogenetics.* Journal of molecular evolution, 1991. **32**(5): p. 443-445.

36. Goloboff, P.A., J.S. Farris, and K.C. Nixon, *TNT, a free program for phylogenetic analysis.* Cladistics, 2008. **24**(5): p. 774-786.