

# Data simulation of tumor phylogenetic trees and evaluation of phylogenetic reconstructing tools

Xinyue Li

**School of Science**

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo October 8, 2017

**Thesis supervisor:**

Prof. Juho Rousu

**Thesis advisors:**

Prof. Veli Mäkinen

D.Sc. Alexandru I. Tomescu

Author: Xinyue Li
Title: Data simulation of tumor phylogenetic trees and evaluation of phylogenetic reconstructing tools
Date: October 8, 2017      Language: English      Number of pages: 4+45
Department of Computer Science
Professorship: Bioinformatics
Supervisor: Prof. Juho Rousu
Advisors: Prof. Veli Mäkinen, D.Sc. Alexandru I. Tomescu
<p>Tumor heterogeneity describes that a tumor usually contains more than one type of cells which are called clones. Clones in a tumor have distinct morphological and physiological features such as genetic variations. Different clones display different sensitivities to cytotoxic drugs, and tumor heterogeneity can add complexity to understand tumor composition and pose challenges for the development of successful therapies. Thus, studying tumor heterogeneity can guide tumor therapies for individual patient and enhance our understanding of inter-clonal functional relationships during therapies, which could be benefit to personalized and efficient treatments.</p> <p>Heterogenetic tumor development is an evolutionary process. There exists an evolutionary relationship among the clones of a heterogenetic tumor and the relationship can be described by an phylogenetic tree. Computational tools have been increasingly important to study tumor heterogeneity because of their time and economic efficiency. Such tools usually take as input the genetic variability data produced by high-throughput sequencing technologies, then output clonal composition of a tumor and reconstruct the polygenetic tree of it.</p> <p>In this thesis, we simulated a large amount of datasets consisting of tumor phylogenetic trees with varying properties and used the datasets to evaluate five recent and popular tumor phylogenetic reconstructing computational tools. We found relatively large differences for performance among those tools and also their strengths and shortcomings, respectively. We left as future work improvement of the data simulation methods and exploration of tool parameters for possibly more beneficial results.</p>
Keywords: bioinformatics; tumor heterogeneity; phylogenetic trees; phylogenetic reconstructing tools; evaluation

## Preface

Thank you professor Juho Rousu for being my supervisor and the guidance. Thank you professor Veli Mäkinen for the interesting thesis topic and the patience for instructions. Thank you Alexandru I. Tomescu for your very much patience for guiding me for all the working details and the thesis revision to every single word. And thank you Lu Cheng for helping me to find this thesis position. And also I thank people around me for their continuous supporting.

Otaniemi, October 8, 2017

Xinyue Li

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Preface</b>	<b>iii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Thesis structure . . . . .	2
<b>2 Biological Background for Our Data Simulation Algorithm</b>	<b>3</b>
2.1 Genetic variability . . . . .	3
2.2 Tumor evolution . . . . .	6
2.3 Sequencing technology and <i>VAF</i> value . . . . .	6
<b>3 Data Simulation of Tumor Plynogenetic Trees</b>	<b>8</b>
3.1 Trees . . . . .	8
3.2 Overview of the data simulation algorithms of the evaluated tools . .	8
3.3 Data simulation . . . . .	11
3.3.1 Non-uniformity of data simulation of Ancestry . . . . .	12
3.3.2 Data simulation with Prüfer sequence . . . . .	21
<b>4 Evaluation of the Tumor Phylogenetic Reconstructing Tools with Simulated Data</b>	<b>30</b>
4.1 Evaluation criteria . . . . .	30
4.2 Overview of the evaluation criteria of the evaluated tools . . . . .	31
4.3 Evaluation results . . . . .	33
4.3.1 Predicted tree of the tools for one of our simulated data . . .	33
4.3.2 Performance of the tools on our evaluation criteria . . . . .	34
<b>5 Discussion</b>	<b>40</b>
<b>6 Summary</b>	<b>41</b>
<b>References</b>	<b>42</b>

# 1 Introduction

Tumor is formed from the abnormal cell growth with the potential to invade or spread to other parts of the body. Malignant tumors are usually called cancers. Cancer has multiple causes such as genetic variations, environmental pollution or poor lifestyle choices. A tumor may consist of subpopulations of cells with distinct genomic alterations, this phenomenon is called tumor heterogeneity. Tumor heterogeneity is likely to have implications for cancer therapeutics and biomarker discovery, particularly in the era of targeted treatment [1]. Current therapies treat cancer as a homogenous disease [2]. Targeted drugs have been developed against single or multiple subpopulations of cells with mutated oncogenes that they target, while those subpopulations without corresponding mutations are unaffected [1]. The pre-existing untargeted subpopulations may expand and maintain tumor progression after the drug therapies [1]. For example, the enrichment of tumorigenic cells in colorectal carcinoma (CRC), breast cancer, and glioblastomas (GBMs) has been observed after irradiation or cyclophosphamide treatment [2]. Therefore, studying tumor growth process and its heterogeneity has profound effects on cancer diagnosis and therapies.

Tumors can be benign, in situ, malignant, and of uncertain or unknown behavior [21]. Benign tumors include uterine fibroids and melanocytic nevi. They are circumscribed and localized and do not transform into cancers [22]. Potentially malignant tumors include carcinoma in situ (CIS). They do not invade and destroy other tissues but may transform into cancers [23]. Malignant tumors are commonly called cancers. They invade and destroy the surrounding tissue, may form metastases and, if untreated or unresponsive to treatment, will prove fatal [23].

Tumor heterogeneity describes that a tumor contains more than one type of cells in it. Different types of cells within a tumor has distinct morphological and physiological features, such as expression of cell surface receptors, proliferative and angiogenic potential. Tumor heterogeneity can occur both between tumors (inter-tumor heterogeneity) or within tumors (intra-tumor heterogeneity). It is widely accepted that tumor development is an evolutionary process [20], and spontaneous tumors usually originate from a single cell and expand to a group of cells which form a mass.

There are two sources for tumor heterogeneity. One source is cancer stem cells, which are non-heritable and the other source is clonal evolution, which is heritable [20]. The concept of cancer stem cells states that the growth and progression of many tumors are driven by a small fraction of cells, the majority of cells in a tumor are the products of abnormal differentiation of cancer stem cells [20]. Thus, to characterize and eliminate the malignant cells in tumors, it is necessary to focus on the small fraction of tumorigenic cells [24]. The concept of clonal evolution states that a tumor arises from a genetically normal cell which evolves to a large amount of cells. In this evolution, random mutations are constantly produced and the tumor finally contains billions of malignant cells that have accumulated large numbers of mutations [25]. Tumor evolution is depicted as a succession of clonal expansion rounds, in which every new round is driven by an additional mutational event [20].

One is a linear model of clonal succession, where progressive sequentially ordered mutations drive linear succession of rounds of clonal expansion, and result in clonal expansion [20]. The other one is multi-clonal model of tumor progression, in which a single cell is expanded into multiple subclones through a splitting mechanism [26]. This model is more associated with tumor heterogeneity than the linear model. The acquired mutations result in an increased genomic instability with each successive generation [27].

Heterogenetic tumors which are tumors consisting of multiple clones can display different sensitivities to cytotoxic drugs among different clones. Furthermore, the level of heterogeneity can itself be used as a biomarker since more heterogenetic tumors may be more likely to contain treatment-resistant clones [28]. The reasons for different sensitivities could be the interactions between clones which may inhibit or alter therapeutic efficacy [20]. Tumors with high heterogeneity have higher probability to be composed of pre-existent clones that are resistant to therapies and may result in therapy failure [20]. The future treatment of tumors relies on personalized strategies designed to target tumorigenic cell populations present in an individual patient [29]. Tumor heterogeneity is a cause to drug resistance and, thus, a prominent contributor to therapeutic failure [29]. Tumors can achieve drug resistance in various ways simultaneously, thus targeting a single resistance mechanism to overcome therapeutic failure may limit the benefit of targeted therapies [30]. Therefore, tumor heterogeneity can add complexity to understand tumor development and pose challenges for the development of successful therapies [29]. Studying tumor heterogeneity can guide subsequent therapy for individual patients and enhance our understanding of inter-clonal functional relationships during therapy, which could be helpful with personalized and efficient therapies [30].

To study tumor heterogeneity, many effective computational tools have been developed to analyze tumor clonal information and its evolutionary history. The tools usually take as input the genetic variability data produced by the moderately-priced high-throughput sequencing technologies, and produce output, such as the clonal composition of a tumor and the ancestral relationship among clones. This information is important to understand the tumor development and help with efficient treatment developments.

## 1.1 Thesis structure

In this thesis, our goal is to evaluate and compare some computational tools which could analyze tumor clonal composition information and the evolutionary relationship among clones. For evaluation, we simulated datasets consisting of tumor phylogenetic trees and set multiple evaluation criteria. We ran the selected tools on our simulated datasets and computed their performance on our evaluation criteria for comparison. Section 2 of the thesis introduces the biological background needed to understand the tumor evolutionary process and how the data is being generated. Section 3 describes our data simulation algorithm. Section 4 presents our evaluation criteria and the evaluation results of the selected tools. Finally, Section 5 indicates some points for further studies.

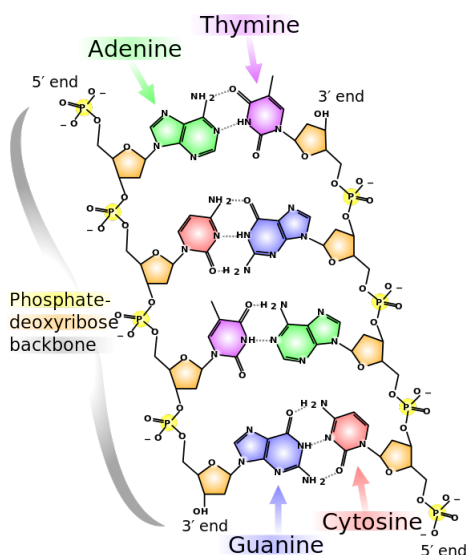


Figure 1: DNA double helix. [31]

## 2 Biological Background for Our Data Simulation Algorithm

In this thesis, we generated datasets of tumor phylogenetic trees. In this section, we briefly introduce biological background related to tumor phylogenetic trees which form the biological basis of our data simulation algorithm.

### 2.1 Genetic variability

*DNA* is a biological molecule polymerized by nucleotides. There are four kinds of nucleotides in DNA: adenine (A), thymine (T), cytosine (C) and guanine (G). DNA is the basis of amino acid sequence which constitutes protein. A DNA molecule consists of two strands. They are antiparallel to each other to form a structure of double helix. Each type of nucleotide on one strand is connected with another type of nucleotide on the other strand: A with T; C with G (Figure 1) [3]. This is known as the rule of base pairing.

DNA replication is the process of producing two identical DNA molecules from the original DNA molecule. When the replication starts, the two strands of a DNA molecule are separated from each other and each strand serves as a template to make its counterpart. A nucleotide at each position of a strand connects with another type of nucleotide based on the rule of base pairing to synthesize the counterpart of this strand. After the replication, the original DNA molecule turns into two identical molecules (Figure 2) [3].

A gene is a region of DNA and is the molecular unit of heredity [3]. There are multiple genes on DNA with different functions. Mutation is a permanent change of the nucleotide sequence of the genome. It can come from the DNA replication process when wrong nucleotides are connected to specific base positions. There are

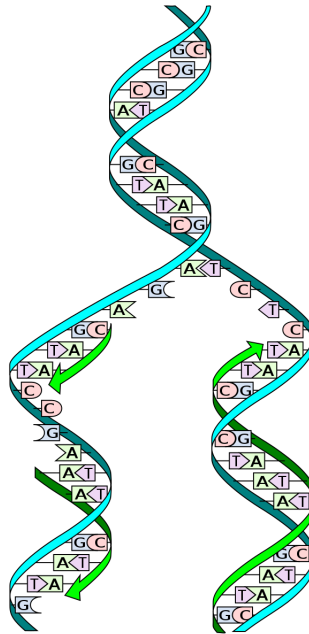


Figure 2: DNA replication. [31]

original sequence:

ACTTGGTCAGAAATTCCCAGGTGTCA

point mutation:

ACTTGGTCATAAATTCCCAGGTGTCA

Figure 3: Point mutation.[9]



insertion:

ACTTGGTCAGAATTCCCAGGTGTCA



ACTTGGTCAGATAGGCAATTCCCAGGTGTCA

deletion:

ACTTGGTCAG~~GAATT~~CCCAGGTGTCA

ACTTGGTCACCCAGGTGTCA

reversion:

ACTTGGTCAGAATTCCCAGGTGTCA



ACTTGGTC~~TTAAGA~~CCCAGGTGTCA

Figure 4: Structural variation. [10]

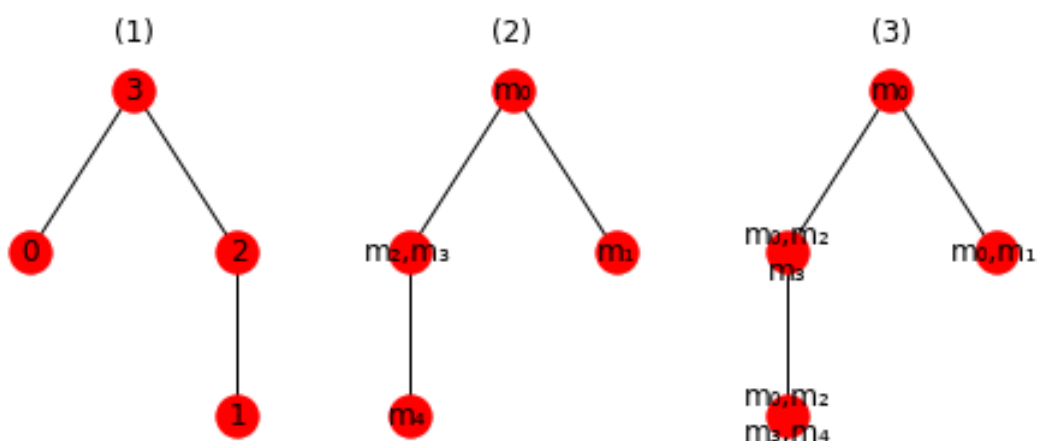
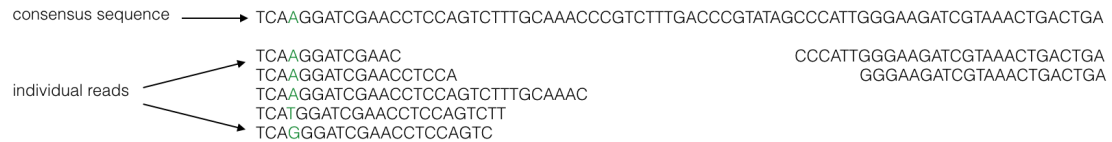
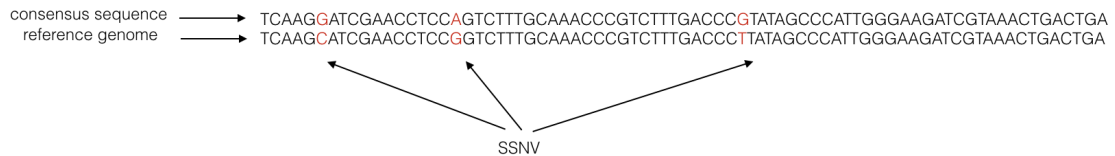


Figure 5: Phylogenetic tree of a tumor

different types of mutations such as single nucleotide mutation (point mutation) (Figure 3) and structural variation (SV) including insertion, deletion, and reversion (Figure 4). Causes leading to mutations might be chemical materials, toxicity or viruses. Mutations in a gene can change the products of it such as a different protein, or prevent the gene from functioning properly [3].



(a) Creating consensus sequence from reads

(b) Comparing consensus sequence with reference genome to detect *SSNV*Figure 6: *SSNV* detection from reads alignment

## 2.2 Tumor evolution

Mutations occur in any cell of the body excluding the germ cells (sperm and egg) are called somatic mutations [6]. The accumulation of somatic mutations during an individual's lifetime can lead to uncontrolled growth of a collection of cells into a tumor [7] and can cause cancer or other diseases [6]. As a result of the accumulation, there will be more than one type of cells in a tumor. Groups of cells with distinct sets of mutations are called clones or a cell population of a tumor. Clones in a tumor are phylogenetically related. Their relationship can be represented by a phylogenetic tree [20]. The phylogenetic tree illustrates evolutionary relationship among clones and the occurrence order of each mutation. For example, in Figure 5, (1) is a phylogenetic tree of a tumor with four clones labeled from 0 to 3, (2) shows new mutations that occurred in each clone during the evolution of this tumor. Each clone will also inherit mutations on the path from the top clone to itself. For example, clone 0 has mutations  $m_0, m_1$ ; clone 1 has mutations  $m_0, m_2, m_3, m_4$ .

## 2.3 Sequencing technology and *VAF* value

DNA sequencing is a method to detect the precise order of nucleotides in a strand of DNA. The next-generation sequencing (NGS) field consists of a number of modern sequencing technologies which allow sequence determination cost and time efficiently. From an input of biological samples, the technologies output short nucleotide sequences (called reads). Reads are then aligned to a reference genome using various alignment algorithms, such as Burrows-Wheeler transform algorithm. After the alignment, a consensus sequence can be created by assembling the overlapping reads (Figure 6). At a position of the consensus sequence, there may be more than one reads type of nucleotides aligned there due to the overlapping of reads (the total number of reads covering a mutation is called read coverage). The nucleotide at that position is determined to be the most common one of the aligned nucleotides. For example, in Figure 6, three adenines (A), one guanine (G) and one thymine (T)

are aligned at the third position of the consensus sequence, then nucleotide at that position is determined to be adenine (A). After creating the consensus sequence, nucleotides in it that are different from the reference genome can be identified and they are the *SSNVs* we need.

From an input of multiple samples from a tumor, we can detect the somatic single nucleotide variation (*SSNV*) in each sample with sequencing technology. The fraction of cells in a sample containing a *SSNV* out of all cells is called the *variant allele frequency* (*VAF*) of a *SSNV* in this sample. We can compute *VAF* values for each *SSNV* in each tumor sample. From an input of *VAF* values of a tumor, many tools have been developed to reconstruct the phylogenetic tree of the tumor.

In this thesis, we generated some datasets of tumor phylogenetic trees. We collect samples from each tree and compute *VAF* values for each *SSNV* in each sample. The *VAF* values are basically the inputs for tools we will evaluate in this thesis. Each tool will output one or several reconstructed phylogenetic trees of the tumor and we then compare the reconstructed tree or trees with the true tree to evaluate the reconstruction accuracy of each tool. We evaluated five phylogenetic tree reconstructing tools: MIPUP [15], LICHeE [16], AncesTree [17], CITUP [18] and Treomics [19].

### 3 Data Simulation of Tumor Plynogenetic Trees

In this work, we simulated datasets consisting of simulated phylogenetic trees. In this section, we first briefly describe concepts related to trees in Section 3.1, then in Section 3.2, we describe the simulation algorithms used in the papers introducing the tools we will evaluate. Our data simulation algorithm is a modification of the simulation and reconstructing algorithms from [17] and is described in Section 3.3.

#### 3.1 Trees

In this section we use the terminology and definitions from [11]. A *tree* is an undirected connected graph which has no cycles. Any two nodes in a *tree* are connected with exactly one path. A *rooted tree* is a *tree* in which one node is set to be *root*. In a *rooted tree*, for a pair of directly connected nodes, the one closer to *root* is called *parent* and the one further to *root* is called *children* of the *parent*. On the path from *root* to a node, all nodes are called *ancestors* of that node. Starting from a node, all nodes reachable by repeatedly proceeding from *parent* to *child* are called *descendants* of that node. A group of nodes with the same *parent* are said to be *siblings*. Nodes without *child* are called *leaves*. Nodes which are not *leaves* are called *internal nodes*. A *subtree* of a *tree* is a *tree* rooted at an *internal node* of that tree. The number of edges incident to a node is called its *degree*.

A labeled tree is a tree with its nodes having distinct labels [12]. The number of unrooted labeled trees of  $n$  nodes is  $n^{n-2}$  [12]. For example, with  $n = 3$ , there are three unrooted labeled trees which are enumerated in Figure 7, and there are nine rooted labeled trees which are enumerated in Figure 8.

#### 3.2 Overview of the data simulation algorithms of the evaluated tools

In this section we give a brief overview of the data simulation algorithms used by the tools that we evaluated.

In LICHeE [16], the simulator (Algorithm 1) starts from a normal clone, which is a clone without mutations, and sets it to be root of a phylogenetic tree. Then iteratively, each clone in the phylogenetic tree can give rise to a new clone which is a child of the clone and also a new node in the phylogenetic tree. The new clone carries a new somatic single nucleotide variation (*SSNV*) with probability  $P_{SSNV}$  or a copy number variation (*CNV*) with probability  $P_{CNV}$ . Each clone can also undergo death with some probability. The simulator runs fifty iterations and produces a phylogenetic tree with several hundreds to thousands of nodes. Then multiple samples are collected from the tree using two different sampling methods. One method is randomized sampling in which one sample is made up of up to five randomly selected nodes from the phylogenetic tree Figure 9. Another method is localized sampling. This method collects all nodes in an entire subtree to form a sample, but subtrees for different samples should be disjoint, which means subtrees for different samples do not share nodes. In both sampling methods, each sample

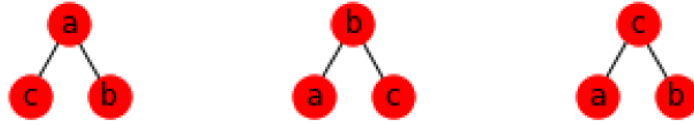


Figure 7: Unrooted labeled trees with 3 nodes

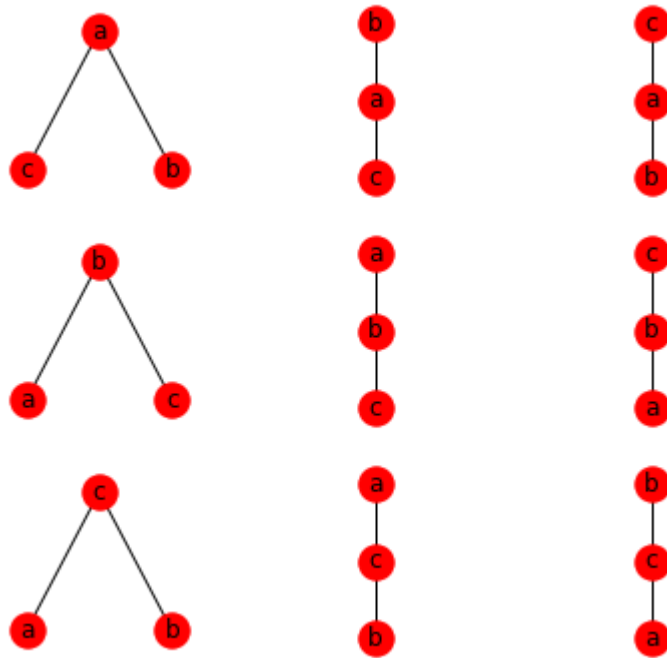


Figure 8: Rooted labeled trees with 3 nodes

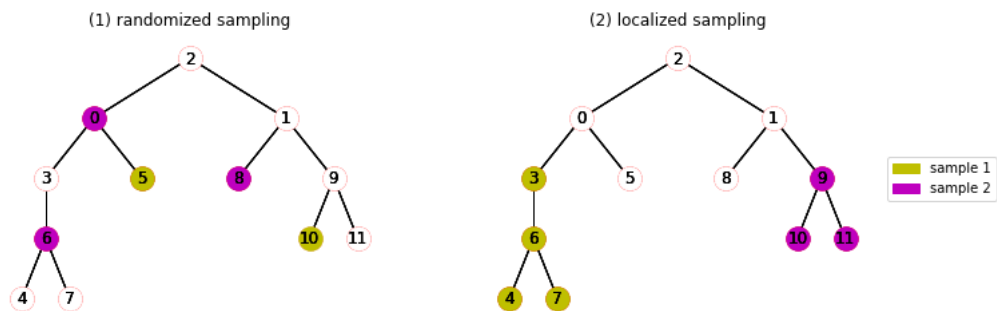


Figure 9: Sampling methods of the paper presenting LICHeE

then collects a fraction of cells from each clone. The algorithm then computes  $VAF$  values for each mutation in each sample. These are the true  $VAF$  values, which are  $VAF$  values without noise. The algorithm then adds sequencing and sampling noise to them, as follows. For sample  $s$  and mutation  $i$ , it samples a value from a binomial distribution  $B(n, p)$  where  $p$  is the true  $VAF$  value of mutation  $i$  in sample  $s$  and  $n$  is a given read coverage. These values are considered to simulate  $VAF$  values obtained from real samples. LICHeE takes as input these values.

---

**Algorithm 1:** Data simulation algorithm of LICHeE

---

```

1 Let  $a$  be the total number of reads;
2 Let  $n\_mutations$  be the number of mutations;
3 Let  $n\_samples$  to be number of samples;
4 Let  $VAF$  be an empty matrix of size  $n\_samples \times n\_mutations$  to store the
   true  $VAF$ s;
5 Let  $\widehat{VAF}$  be an empty matrix of size  $n\_samples \times n\_mutations$  to store the
   noisy  $VAF$ s;
6 Let  $sample\_i$  = a list of selected nodes of the  $i$ th sample;
7 Let  $mutations$  to be a list of mutations in the  $tree$ ;
8 Set one node with normal cells to be the  $root$  of a tree;
9 for  $i = 1$  to  $50$  do
10   for  $x$  in existing nodes in the tree do
11     | add a new node to  $x$  with  $P_{SSNV} = 0.15$  or  $P_{CNV} = 0/0.1/0.18$ ;
12   end
13 end
14 for  $j = 1$  to  $n\_samples$  do
15   |  $sample\_i$  = up to 5 randomly selected nodes or all nodes in one subtree;
16 end
17 for  $i = 1$  to  $n\_samples$  do
18   for  $j = 1$  to  $n\_mutations$  do
19     |  $VAF_{ij}$  = the fraction of cells containing the  $mutations[j]$  out of all the
       | cells in  $sample\_i$ ;
20     |  $\widehat{VAF}_{ij}$  = a random number from binomial distribution  $Bin(a, VAF_{ij})$ ;
21   end
22 end

```

---

In AncesTree [17], the simulation algorithm (Algorithm 2) assigns one hundred mutations into ten clones uniformly at random. Then it randomly picks a clone to be the root of its phylogenetic tree. Then iteratively, the algorithm randomly picks a node from the remaining nodes and adds it as child to one of the existing nodes of the tree until there are no nodes left. Then the algorithm collects samples from the tree to compute  $VAF$  values and the description of those process is short and a bit confusing. The algorithm creates a usage matrix  $U$  row by row where each row represents the usage of a sequenced sample. Usage was determined by first selecting the number  $c$  of clones mixed in each sample by uniformly at random selecting a value

between 1 and 4. Then, usage was determined by randomly sampling a value from the  $c$  simplex and applying the first  $c$  values to  $c$  randomly selected clones. The algorithm used rejection sampling over this whole process to ensure that only simulations where all mutations were included in at least two samples were created. The simulation process described above implicitly creates a pair of matrices  $B$  and  $U$ . Then the VAF value matrix is  $F = \frac{1}{2}UB$ . AncesTree requires read counts for reference alleles and alternate alleles to be its inputs so it simulate read counts, as follows. For sample  $s$  and mutation  $i$ , the algorithm samples the number of reads containing mutation  $i$  for sample  $s$  as  $n_{si} \sim Poiss(a)$ , where  $a$  is a given read coverage; then it samples the number of reads containing the variant allele as  $x_{si} \sim Binomial(n_{si}, f_{si})$ . The number of reads containing the reference allele is  $n_{si} - x_{si}$ .

---

**Algorithm 2:** Data simulation algorithm of AncesTree

---

```

1 Let  $n_{nodes} = 10$ ;
2 Divide 100 mutations uniformly at random into 10 nodes;
3 Randomly set one node to be the root of the tree;
4 while  $n_{nodes} > 0$  do
5   | randomly selected a node and add it to one of the existing nodes in the tree;
6   |  $n_{nodes} = n_{nodes} - 1$ ;
7 end
8 Creating matrices  $U$  and  $B$ ;
9  $F = \frac{1}{2} \times U \times B$ ;
10 Let  $read\_coverage = r$ ;
11 for  $i = 0$  to 99 do
12   | for  $p = 0$  to  $n_{sample} - 1$  do
13     | |  $n[i, p]$  = a random number from a poisson distribution  $Poiss(r)$ ;
14     | |  $x[i, p]$  = a random number from a binomial distribution
15     | |  $B(n[i, p], F[p, i])$ ;
16   | end
17 end
18  $n\_x = n - x$ ;

```

---

CITUP [18] has a short description of their data simulation algorithm. It creates phylogenetic tree of three to seven clones. The clone frequencies were generated by sampling from a Dirichlet distribution. The algorithm uniformly assigns five hundred mutations into the clones and it also adds sequencing and sampling noise to the simulated data by sampling from a Gaussian distribution.

In Treomics [19], there is no detailed description of their data simulation algorithms.

### 3.3 Data simulation

After exploring the simulation algorithms of the evaluated tools, we decided to use a modification of the simulation and reconstructing algorithms in AncesTree to

generate our data. In our modification, we use a Prüfer sequence [13] to generate phylogenetic tree. The Prüfer sequence (also Prüfer code or Prüfer numbers) is a unique sequence associated with a labeled tree. Given a labeled tree, its Prüfer sequence can be generated by a simple iterative algorithm. Correspondingly, given the Prüfer sequence of a labeled tree, the tree can be generated by an algorithm. After generating phylogenetic trees with Prüfer sequence, we then collect samples and compute  $VAF$  values as done in the reconstructing algorithm of Ancestree. We did not generate trees as in the simulation algorithm of Ancestree (by adding a randomly selected node iteratively) because we found it is unable to generate trees uniformly. In the rest of this section, we prove the non-uniformity of simulation algorithm of Ancestree and then describe our data simulation algorithm with a Prüfer sequence.

### 3.3.1 Non-uniformity of data simulation of Ancestree

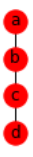
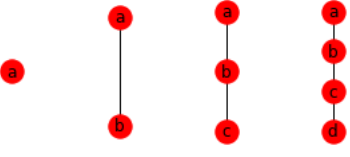

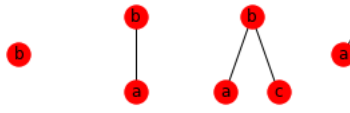
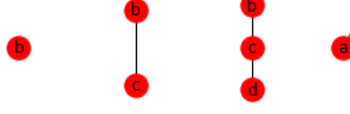
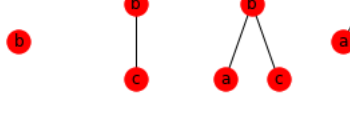

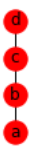
The data simulation algorithm of paper presenting Ancestree is described in the third paragraph in Section 3.2. For a certain number of nodes, there are more than one rooted labeled trees. To prove the non-uniformity of the data simulation algorithm of Ancestree, we need to prove that under the simulation algorithm of Ancestree, the generating probabilities are unequal for different rooted labeled trees. In this section, the proof is started from a simple counterexample which are rooted labeled trees of 4 nodes. We demonstrate all unrooted labeled trees of 4 nodes, and each unrooted labeled tree corresponds to 4 rooted labeled trees by setting each node to be its root. We prove that the probabilities of generating each rooted labeled tree of 4 nodes are unequal. The simple counterexample intuitively demonstrates the non-uniformity of the algorithm, but to state the algorithm is not uniform, the non-uniformity property should not only hold for trees with 4 nodes but also for an arbitrary number of nodes. Thus after the counterexample, we extend the proof to trees with an arbitrary number  $n$  of nodes. In that situation, demonstrating all unrooted labeled trees is too complex because there are too many of them. So we demonstrate two special unrooted labeled trees and their corresponding rooted labeled trees. We show the generating probabilities are unequal for those rooted labeled trees, which can prove the non-uniformity of data simulation algorithm of Ancestree.

We first give a counterexample which is a tree with four nodes (According to Cayley's formula [12], with  $n$  nodes, there are  $n^{n-2}$  unrooted labeled trees. If using an example with three nodes, there are only three trees and if using example with five nodes, there are 125 trees which is too complex to demonstrate. An example with four nodes has 16 trees which might be a suitable example).

All unrooted labeled trees with 4 nodes are shown in Figure 10. With each node to be root, every unrooted labeled tree corresponds to 4 rooted labeled trees. For example, rooted labeled trees for tree (1) in Figure 10 are in Figure 11. Basically, unrooted labeled trees with 4 nodes have two structures: a tree whose nodes have degree at most 2 such as (1)-(12) in Figure 10; a tree whose nodes have degree at most 3 such as (13)-(16) in Figure 10. We will prove that the generating probabilities of different rooted labeled trees of (1) are unequal, and it is easy to prove that for (2)-(12) in a similar way. Then, we prove that the generating probabilities of different



Table 1: Generating probability of rooted labeled trees of tree (1) in Figure 10

Rooted labeled trees of tree (1)	Generating patterns	Probability
rooted labeled tree 1 	pattern 1 	$\frac{1}{144} \times 1 = \frac{1}{144}$
rooted labeled tree 2 	pattern 1  pattern 2  pattern 3 	$\frac{1}{144} \times 3 = \frac{3}{144}$
rooted labeled tree 3 	Similar with rooted labeled tree 2	$\frac{3}{144}$
rooted labeled tree 4 	Similar with rooted labeled tree 1	$\frac{1}{144}$

rooted labeled trees of (13) are unequal, and it is easy to prove that for (14)-(16) are unequal in a similar way.

**Lemma 1.** *Under the simulation algorithm of the paper presenting AncestryTree, the generating probabilities are unequal for different rooted labeled trees of (1) in Figure 10.*

*Proof.* All rooted labeled trees of (1) in Figure 10 are shown in the first column of Table 1. Based on the simulation algorithm of AncestryTree, we give patterns of generating each rooted labeled tree (second column of Table 1). At rooted labeled tree 2, for example, the first selected node has to be node  $b$  because AncestryTree sets the first selected node to be its root. The second selected node can be  $a$  or  $c$ . If it

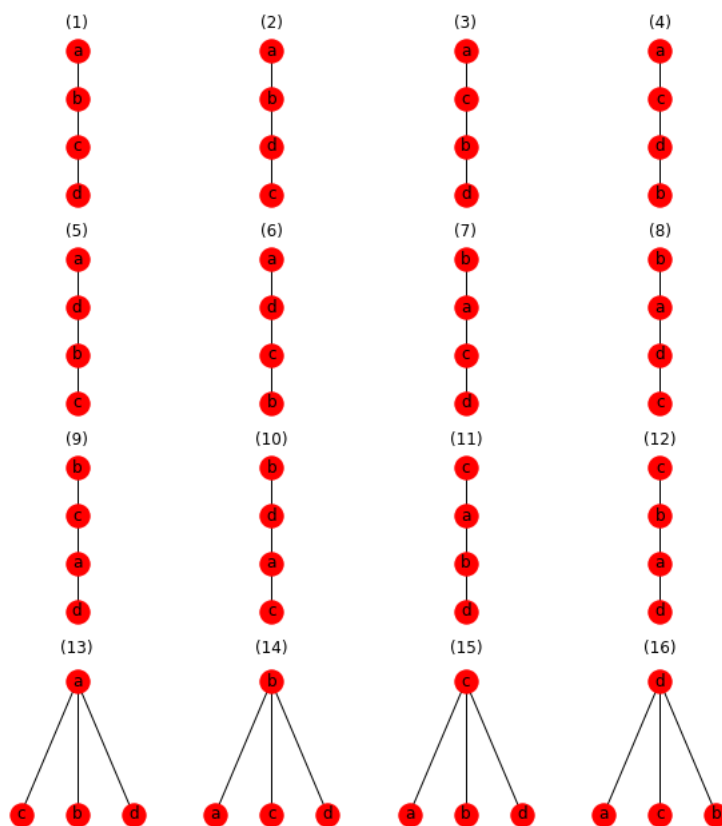


Figure 10: Unrooted labeled trees with 4 nodes

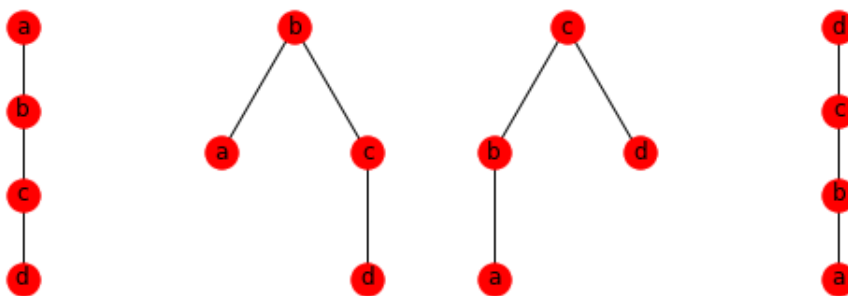


Figure 11: Rooted labeled trees for (1) in Figure 10, with the root shown as top-most node

is  $a$ , the third selected node has to be  $c$  and the fourth selected node has to be  $d$ . If the second selected node is  $c$ , the third selected node can be  $d$  or  $a$ . And then the fourth selected node is  $a$  or  $d$ , respectively. We can see there are 3 patterns of generating rooted labeled tree 2.

For pattern 1 of rooted labeled tree 2, the probability of selecting node  $b$  at the first iteration is  $\frac{1}{4}$ , the probability of selecting node  $a$  at the second iteration is  $\frac{1}{3}$ . The probability of selecting node  $c$  at the third iteration is  $\frac{1}{2}$ , but node  $c$  has to be connected with node  $b$  instead of  $a$  with another probability  $\frac{1}{2}$ . The probability of selecting node  $d$  at the last iteration is 1, but it has to be connected with node  $c$  with another probability  $\frac{1}{3}$ . So the probability of pattern 1 is  $\frac{1}{4} \times \frac{1}{3} \times (\frac{1}{2} \times \frac{1}{2}) \times (1 \times \frac{1}{3}) = \frac{1}{144}$ . It is easy to prove that the probability of any pattern for any rooted labeled tree in Table 1 is the same. Thus we obtained the probability of generating each rooted labeled tree for (1) (third column of Table 1). Probabilities in the third column of Table 1 are unequal, thus, Lemma 1 is proved. ■

**Lemma 2.** *Under the simulation algorithm of the paper presenting AnceTree, the generating probabilities are unequal for different rooted labeled trees of (13) in Figure 10.*

*Proof.* All rooted labeled trees of (13) in Figure 10 are shown in the first column of Table 2. Based on simulation algorithm of AnceTree, we give patterns of generating each rooted labeled tree. At rooted labeled tree 2, for example, the first selected node has to be node  $b$ . The second selected node has to be  $a$ . The third selecting node can be  $c$  or  $d$  and then the fourth node is  $d$  or  $c$ , respectively.

For pattern 1 of rooted labeled tree 2, the probability of selecting node  $b$  at the first iteration is  $\frac{1}{4}$ , the probability of selecting node  $a$  at the second iteration is  $\frac{1}{3}$ . The probability of selecting node  $c$  at the third iteration is  $\frac{1}{2}$ , but node  $c$  has to be connected with node  $a$  instead of  $b$  with another probability  $\frac{1}{2}$ . The probability of selecting node  $d$  at the last iteration is 1, but it has to be connected with node  $a$  with another probability  $\frac{1}{3}$ . So the probability of pattern 1 is  $\frac{1}{4} \times \frac{1}{3} \times (\frac{1}{2} \times \frac{1}{2}) \times (1 \times \frac{1}{3}) = \frac{1}{144}$ . It is easy to prove the probability of any pattern for any rooted labeled tree in Table 2 is the same. So we obtained the probability of generating each rooted labeled tree for (13) (third column of Table 2). Probabilities in the third column of Table 2 are unequal, thus, Lemma 2 is proved. ■

**Theorem 1.** *The generating probabilities under the simulation algorithm of the paper presenting AnceTree are unequal for different rooted labeled trees with 4 nodes*

*Proof.* Lemma 1 proved that the generating probabilities are unequal for different rooted labeled trees of (1) in Figure 10, we can prove that the generating probabilities of different rooted labeled trees of (2)-(12) are unequal in a similar way with (1). Lemma 2 proved that the generating probabilities are unequal for different rooted labeled trees of (13) in Figure 10, we can prove that the generating probabilities of different rooted labeled trees of (14)-(16) are unequal in a similar way with (13). Thus, we can conclude that the generating probabilities are unequal for different rooted labeled trees with 4 nodes, which proved Theorem 1. ■

When looking more closely, for (1) in Figure 10, the probability of generating any of its rooted labeled tree that rooted at the first selected node is  $\frac{8}{144}$  (sum up probabilities in third column of Table 1). Since the first selected node can be any of its node,  $\frac{8}{144}$  is also the probability of generating (1), which is an unrooted labeled tree whose nodes have degree at most 2. Similarly, for (13) in Figure 10, the probability of generating any of its rooted labeled tree that rooted at the first selected node is  $\frac{12}{144}$  (sum up probabilities in third column of Table 2), and  $\frac{12}{144}$  is also the probability of generating (13), which is an unrooted labeled tree whose nodes have degree at most 3. It is easy to show that trees with an arbitrary  $n$  of nodes have similar properties. Thus we have the following corollaries:

**Corollary 1.** *With an arbitrary  $n$  of nodes, let  $T$  be an unrooted labeled tree whose nodes have degree at most 2, let  $T'$  be any of the rooted labeled tree of  $T$ . Under the simulation algorithm of paper presenting Ancestree, let  $P(T)$  be the generating probability of  $T$  and let  $P(T')$  be the generating probability of  $T'$ , we have:  $P(T) = P(T')$*

**Corollary 2.** *With an arbitrary  $n$  of nodes, let  $T$  be an unrooted labeled tree whose nodes have degree at most  $n - 1$ , let  $T'$  be any of the rooted labeled tree of  $T$ . Under the simulation algorithm of paper presenting Ancestree, let  $P(T)$  be the generating probability of  $T$  and let  $P(T')$  be the generating probability of  $T'$ , we have:  $P(T) = P(T')$*

Then we extend the proof to trees with an arbitrary number  $n$  of nodes. If  $n$  is large, there are a large number of different unrooted labeled trees. Each one corresponds to  $n$  rooted labeled trees. Enumerating all of them and computing the probability of each might be too complex. So we give two special cases:

- case 1: unrooted labeled trees with an arbitrary number  $n$  of nodes and whose nodes have degree at most 2, similar with (1)-(12) in Figure 10.
- case 2: unrooted labeled trees with an arbitrary number  $n$  of nodes and whose nodes have degree at most  $n - 1$ , similar with (13)-(16) in Figure 10.

Intuitively, a tree of case 1 is to permutate all nodes to be in a "line". With  $n$  nodes, there are  $n!$  permutations, but a node sequence can be read from both sides, for example:  $abcdef$  and  $fedcba$  are different permutations but actually the same tree. So there are totally  $\frac{n!}{2}$  different trees of case 1. For case 2, we can think of a tree as a "star" with all the other nodes connected to a center node. So with  $n$  nodes, there are  $n$  different trees in case 2 by setting each node to be the center node.

We demonstrate an arbitrary tree of case 1 (tree(1) of Figure 12), and it corresponds to  $n$  rooted labeled trees by setting each node to be root (first column in Table 3). Then we show the generating probabilities are unequal for different rooted labeled trees for this arbitrary tree of case 1. Similarly, we demonstrate an arbitrary tree of case 2 (tree(2) of Figure 12), and its corresponding  $n$  rooted labeled trees (first column in Table 4), and show the generating probabilities are unequal for different rooted labeled trees for this arbitrary tree of case 2.

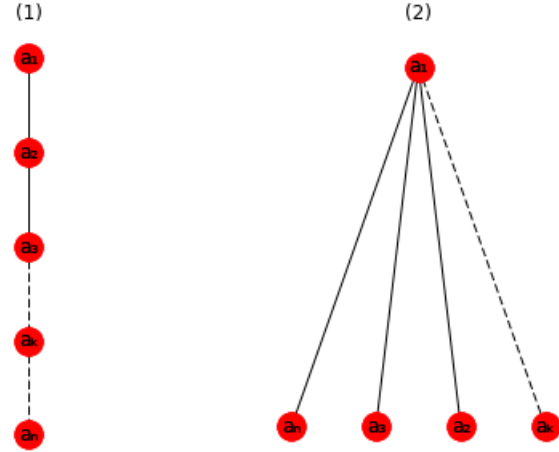


Figure 12: The arbitrary trees of case 1 and case 2

**Lemma 3.** *Under the simulation algorithm of the paper presenting AncestryTree, the generating probabilities are unequal for different rooted labeled trees of tree(1) in Figure 12.*

*Proof.* The patterns of generating tree (1), and the corresponding probabilities, are shown in Table 3. At rooted labeled tree 2, for example, the first selected node should be  $a_2$ , the second selected node can be  $a_1$  or  $a_3$ . If it is  $a_1$ , the third selected node can only be  $a_3$ , the fourth selected node can only be  $a_4$ , and so on. If the second selected node is  $a_3$ , the third selected node can be  $a_1$  or  $a_4$ , and so on. Thus, the total number of patterns of generating rooted labeled tree 2 is  $2 + \underbrace{1 + 1 + \dots + 1}_{n-3} = n - 1$ . For each pattern in rooted labeled tree 2, the probability

of selecting the first node is  $\frac{1}{n}$ , the probability of selecting the second node is  $\frac{1}{n-1}$ , the probability of selecting the third node is  $\frac{1}{n-2}$  and it has to be connected to only one node in the tree to form a specific pattern, so with another probability  $\frac{1}{2}$ , and so do the following nodes. Thus, the probability of each pattern of rooted labeled tree 2 is  $\frac{1}{n} \times \frac{1}{n-1} \times (\frac{1}{n-2} \times \frac{1}{2}) \times (\frac{1}{n-3} \times \frac{1}{3}) \times \dots \times (1 \times \frac{1}{n-1}) = \frac{1}{n! \times (n-1)!}$ . Thus, the probability of generating rooted labeled tree 2 is  $\frac{1}{n! \times (n-1)!} \times (n - 1) = \frac{1}{n! \times (n-2)!}$ . We can verify this result with the previous example of 4 nodes. Indeed, for  $n = 4$ , we have  $\frac{1}{4! \times 2!} = \frac{1}{48} = \frac{3}{144}$ , which is the probability of generating rooted labeled tree 2 in Table 1.

Probabilities of other rooted labeled trees for tree (1) can be computed in similar ways, we omitted them since they are too complex to demonstrate. ■

**Lemma 4.** *Under the simulation algorithm of the paper presenting AncestryTree, the generating probabilities are unequal for different rooted labeled trees of tree(2) in Figure 12.*

*Proof.* The patterns of generating tree (2), and the corresponding probabilities, are

shown in Table 4. At rooted labeled tree 2, for example, the first selected node should be  $a_2$ , the second selected node should be  $a_1$ , the third selected node can be  $a_k$  ( $k$  is from 3 to  $n$ ). The fourth selected node can be one of  $a_3, \dots, a_{k-1}, a_{k+1}, \dots, a_n$ , and so do with following selected nodes. So the number of patterns of generating rooted labeled tree 2 is  $2 \times 3 \times 4 \times \dots \times (n-3) \times (n-2) = (n-2)!$ . For each pattern in rooted labeled tree 2, the probability of selecting the first node is  $\frac{1}{n}$ , the probability of selecting the second node is  $\frac{1}{n-1}$ , the probability of selecting the third node is  $\frac{1}{n-2}$  and it has to be connected to only one node in the tree to form a specific pattern, so with another probability  $\frac{1}{2}$ , and so do with the following nodes. Thus, the probability of each pattern of rooted labeled tree 2 is  $\frac{1}{n} \times \frac{1}{n-1} \times (\frac{1}{n-2} \times \frac{1}{2}) \times (\frac{1}{n-3} \times \frac{1}{3}) \times \dots \times (1 \times \frac{1}{n-1}) = \frac{1}{n! \times (n-1)!}$ . Thus, the probability of generating rooted labeled tree 2 is  $\frac{1}{n! \times (n-1)!} \times (n-2)! = \frac{1}{n! \times (n-1)!}$ . We can verify this result with the previous example of 4 nodes. Indeed, for  $n=4$ , we have  $\frac{1}{4! \times 3} = \frac{1}{72} = \frac{2}{144}$ , which is the probability of generating rooted labeled tree 2 in Table 2.

Probabilities of other rooted labeled trees for tree (2) can be computed in similar ways, we omitted them since they are too complex to demonstrate. ■

The proof can be extended to be more formal than presenting arbitrary trees in case 1 and case 2.

**Lemma 5.** *With an arbitrary  $n$  of nodes, let  $T_1$  be an unrooted labeled tree in case 1, let  $T'_1$  be any of the rooted labeled tree of  $T_1$ ; let  $T_2$  be an unrooted labeled tree in case 2, let  $T'_2$  be any of the rooted labeled tree of  $T_2$ . Under the simulation algorithm of paper presenting AncestryTree, the generating probabilities for  $T'_1$  and  $T'_2$  are unequal, which is  $P(T'_1) \neq P(T'_2)$ .*

*Proof.* we start from considering  $T_1$  and  $T'_1$ . To generate  $T'_1$ , it does not matter which node to select at each iteration and we only need to consider how to connect the selected node to keep the tree whose nodes having degree at most 2 (intuitively, to keep all nodes to be a "line"). Based on simulation algorithm of AncestryTree, the first selected node is root (Figure 13), the second selected node is connected to root. The third selected node can be connected to any of the first two nodes and all nodes will be in a "line". Starting from the fourth selected node, it has to be connected to nodes with degree one to keep all nodes in a "line". So the selected node has two choices with probability  $\frac{2}{k}$ , where  $k$  is the number of existing nodes in the tree. Thus the probability of generating any rooted labeled tree that rooted at its first selected node for any tree in case 1 is  $\frac{2}{3} \times \frac{2}{4} \times \frac{2}{5} \times \dots \times \frac{2}{n-1} = \frac{2^{n-2}}{(n-1)!}$ . Since different trees in case 1 can be regarded as different permutations of  $n$  nodes in a "line", the probability of generating each of them should be equal. With  $n$  nodes, there are  $\frac{n!}{2}$  different trees in case 1. Thus the probability of generating any rooted labeled tree that rooted at its first selected node for one tree in case 1 is  $\frac{2^{n-2}}{(n-1)!} / \frac{n!}{2} = \frac{2^{n-1}}{n! \times (n-1)!}$ , which means  $P(T'_1) = \frac{2^{n-1}}{n! \times (n-1)!}$ .

Then we consider  $T_2$  and  $T'_2$ . To generating  $T'_2$ , the first selected node is root (Figure 14), the second selected node is connected to root. The third selected node can be connected to any of the first two nodes. After its connection, the center node

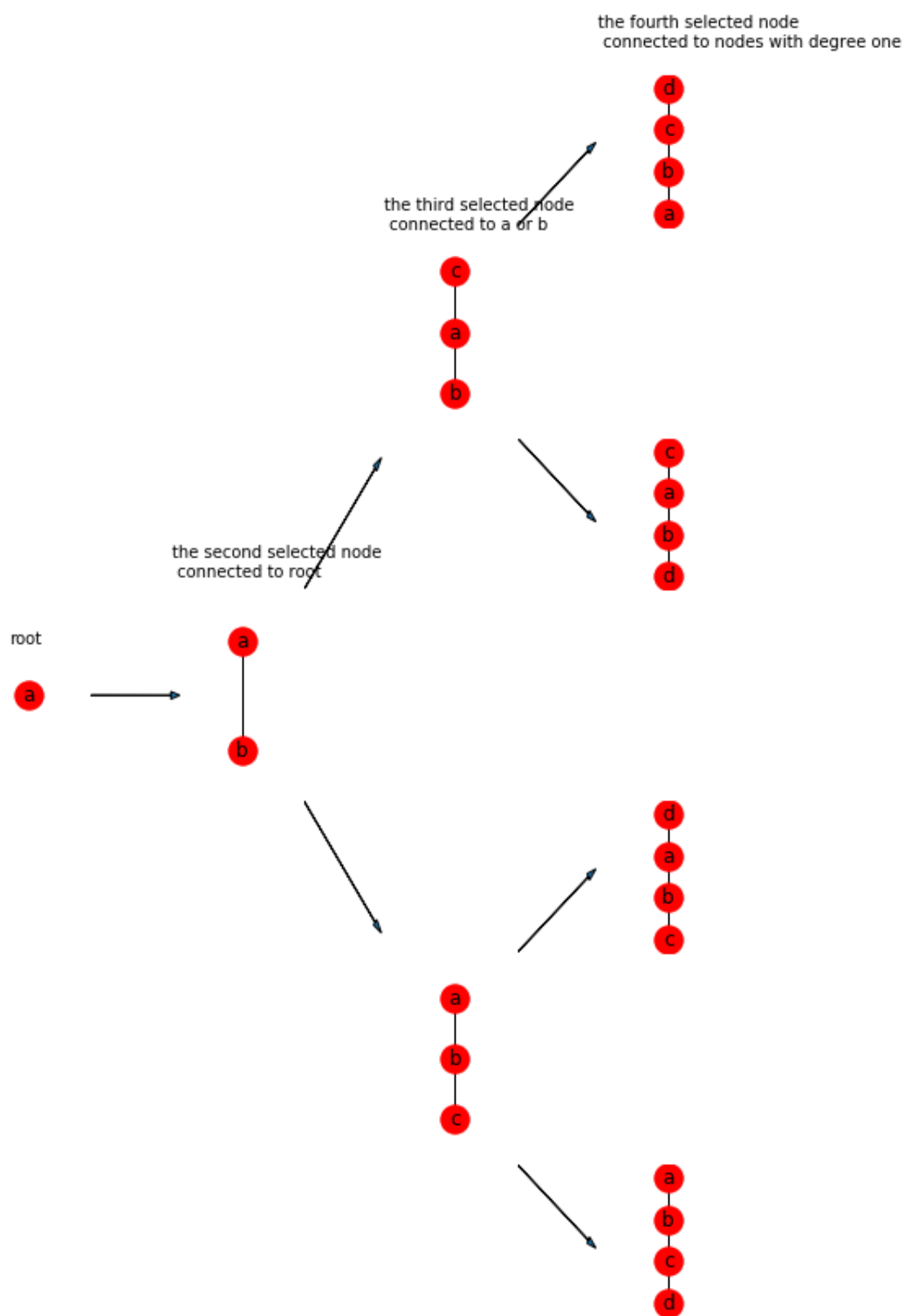


Figure 13: Labeled trees with degree at most 2

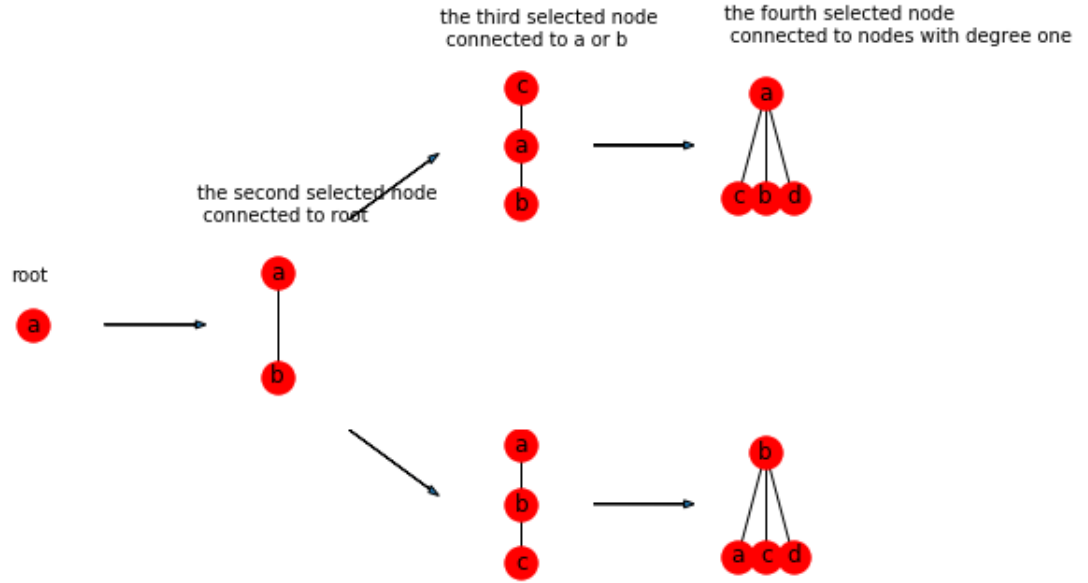


Figure 14: labeled tree with degree at most 3

has been fixed. Starting from the fourth selected node, it has to be connected to the center node with probability  $\frac{1}{k}$ , in which  $k$  is the number of existing nodes in the tree. Thus the probability of generating any labeled tree rooted at its first selected node with degree at most 3 is  $\frac{1}{3} \times \frac{1}{4} \times \cdots \times \frac{1}{n-1} = \frac{2}{(n-1)!}$ . With  $n$  nodes, there are  $n$  different such trees by setting each node to be center node, so the probability of generating any rooted labeled tree that rooted at its first selected node for one tree in case 2 is  $\frac{2}{(n-1)!}/n = \frac{2}{n!}$ , which means  $P(T_2) = \frac{2}{(n-1)!}/n = \frac{2}{n!}$ .

Thus,  $P(T_1) \neq P(T_2)$ , which proved Lemma 5. ■

The proof of Lemma 5 can be verified with the example of 4 nodes. According to Corollary 1,  $P(T_1) = P(T_1')$ , thus,  $P(T_1) = \frac{2^{n-1}}{n! \times (n-1)!}$ . With  $n = 4$ ,  $\frac{2^3}{4! \times 3!} = \frac{1}{18} = \frac{8}{144}$  which is the probability of generating (1) in Figure 10. According to Corollary 2,  $P(T_2) = P(T_2')$ , thus,  $P(T_2) = \frac{2}{n!}$ . With  $n = 4$ ,  $\frac{2}{4!} = \frac{1}{12} = \frac{12}{144}$  which is the probability of generating (13) of Figure 10.

**Theorem 2.** *Under the simulation algorithm of the paper presenting AncestryTree, the generating probabilities are unequal for different rooted labeled trees with an arbitrary number  $n$  of nodes.*

*Proof.* From Lemma 3 and 4 demonstrate that the generating probabilities are unequal for different rooted labeled trees using specific examples. Lemma 5 demonstrates the same conclusion in a more formal way. Thus, we can conclude that with an arbitrary number  $n$  of nodes, the generating probabilities are unequal for different rooted labeled trees, which proved Theorem 2. ■

Thus, the simulation algorithm of the paper presenting AncestryTree is non-uniform.



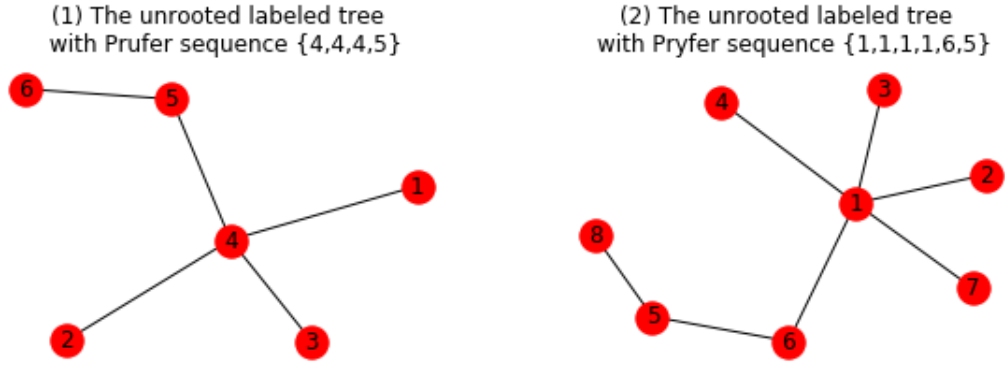


Figure 15: The unrooted labeled trees with their corresponding Prüfer sequences

### 3.3.2 Data simulation with Prüfer sequence

In this subsection, we describe our data simulation algorithm. At the end of this subsection, we use a simplified example to demonstrate every step described below.

Our tree has 10 clones marked from 0 to 9 and 100 mutations marked from  $m_0$  to  $m_{99}$  and are randomly assigned into the clones. We ensure each clone has at least one mutation so there is no empty clone. Then we randomly assign to each clone a cell population size ranging from 100 to 200.

Our phylogenetic tree was randomly generated starting from a Prüfer sequence [12]. A Prüfer sequence is a unique sequence associated with a labeled tree. A labeled tree with  $n$  nodes has such sequence of length  $n - 2$ . Given a random sequence  $a[1], a[2], \dots, a[n]$ , the tree associated with it will have  $n + 2$  nodes. For each node, the algorithm sets its degree to be the number of times the node appears in the sequence plus 1. Then for each number  $a[i]$ , the algorithm finds the lowest-numbered node  $j$  with degree 1, add the edge  $(j, a[i])$  to the tree, and decrease the degrees of  $j$  and  $a[i]$  by 1. Finally, there are two nodes with degree 1 left and the algorithm just add an edge between them. Generating uniformly distributed random sequences and converting them into the corresponding trees is a straightforward method of generating uniformly distributed random labelled trees [14]. Figure 15 demonstrates examples of two unrooted labeled trees with their Prüfer sequences.

To generate phylogenetic trees with  $n = 10$  nodes (Algorithm 3 and 4), we first generate a random sequence of length 8. Each position of the sequence is a random integer from 0 to 9. Then we convert the sequence into an unrooted labeled tree with 10 nodes with Prüfer code and randomly pick a node to be its root. Then, each clone will inherit mutations from all its parent nodes.

Then we collect some samples from the tree. We randomly select 2 to 4 clones to be a sample. The *mutations* in each sample will be the union of mutations in clones it has; the *cell population* size of this sample will be the sum of cell population size of its clones. Then we compute *VAF* values as did in AncesTree. Our tree has 10 clones, thus assuming we collect  $s$  samples from a tree, we first create a  $s \times 10$  usage matrix  $U$ . Each entry  $U_{si}$  is the fraction of the cells belonging to clone  $i$  in

---

**Algorithm 3:** Data Simulation with a Prüfer sequence
 

---

**Data:** The number of samples:  $n_{sample}$

**Result:**  $F_{unpack}$

- 1  $a = [a_0, \dots, a_7]$ , in which  $a_i =$  a random number between 0 and 9;  
 $mutations = [0, 1, \dots, 99]$ ;
- 2 Divide  $mutations$  into 10 sublists:  $l_0, l_1, \dots, l_9$ ;
- 3  $clone =$  empty dictionary;
- 4  $cell\_size =$  empty dictionary;
- 5 **for**  $i = 0$  to 9 **do**
- 6  $clone[i] = l_i$ ;
- 7  $cell\_size[i] =$  a random number between 100 and 200;
- 8 **end**
- 9 Covert  $a$  to a labeled tree with Prüfer code;
- 10 Let  $root =$  a random number between 0 and 9;
- 11 Let  $sample =$  empty dictionary;
- 12 **for**  $i = 0$  to  $n_{sample} - 1$  **do**
- 13  $sample[i] = [c_0, c_1, \dots, c_k]$ ,  $k$  is between 2 and 4,  $c_k$  is a randomly number  
between 0 and 9;
- 14  $sample\_mutations = clone[c_0] + clone[c_1] + \dots + clone[c_k]$ ;
- 15  $sample\_cell\_size = cell\_size[c_0] + cell\_size[c_1] + \dots + cell\_size[c_k]$ ;
- 16 **end**
- 17 Let  $U =$  a empty matrix of size  $n_{sample} \times 10$  ;
- 18  $B =$  a empty matrix of size  $10 \times 10$ ;
- 19 **for**  $i = 0$  to  $n_{sample} - 1$  **do**
- 20 **for**  $j = 0$  to 9 **do**
- 21  $U[i, j] = \frac{cell\_size[i]}{sample\_cell\_size[i]}$ ;
- 22 **end**
- 23 **end**
- 24 **for**  $i = 0$  to 9 **do**
- 25 **for**  $j = 0$  to 9 **do**
- 26 **if**  $i$  is a descendant of  $j$  in the converted tree **then**
- 27  $B[i, j] = 1$ ;
- 28 **else**
- 29  $B[i, j] = 0$ ;
- 30 **end**
- 31 **end**
- 32 **end**
- 33  $F = \frac{1}{2} \times U \times B$ ;
- 34 Let  $F_{unpack} =$  a empty matrix of size  $n_{sample} \times 100$ ;
- 35 **for**  $i = 0$  to 9 **do**
- 36  $m = clone[i]$ ;
- 37 **for**  $j$  in  $m$  **do**
- 38 **for**  $s = 0$  to  $n_{sample} - 1$  **do**
- 39  $F_{unpack}[s, j] = F[s, i]$ ;
- 40 **end**
- 41 **end**
- 42 **end**

---

sample  $s$ . Then we create a  $10 \times 10$  clonal matrix  $B$ . Each row of  $B$  represents a clone, each column represents the group of new mutations occurred in a clone. We set each entry  $B_{ij}$  to be 1 if clone  $i$  (after mutation inheritance) contains the group of new mutations occurred in clone  $j$ ; and set  $B_{ij}$  to be 0 if not. Then the  $VAF$  value matrix  $F = \frac{1}{2} \times U \times B$ . Thus  $F$  is  $s \times 10$  in which  $F_{si}$  is the  $VAF$  value of the group of new mutations occurred in clone  $i$  in sample  $s$ . Since we have 100 mutations, we unpack  $F$  to be  $s \times 100$  by simply repeating column  $i$  to be  $k$  columns, where  $k$  is the number of new mutations occurred in clone  $i$ . After unpacking,  $F_{si}$  is the  $VAF$  value of mutation  $i$  in sample  $s$ .

Value  $F_{si}$  is the base of the input for the tools we will evaluate. However, each tool has its specific input format. To ensure data consistency among tools, we did some additional operations. For sample  $s$  and mutation  $i$ , we draw the number of reads containing mutation  $i$  in sample  $s$  from a Poisson distribution as  $n_{si} \sim Poiss(n)$ , where  $n$  is the given read coverage. Then we draw the number of reads containing the variant allele from a binomial distribution as  $x_{si} \sim B(n_{si}, F_{si})$ . These sampling procedures also add noise to the simulated data and we can control the noise with read coverage. Then the number of reads containing the reference allele is  $n_{si} - x_{si}$ . Here  $x_{si}$  and  $n_{si}$  are inputs of tool AncesTree,  $x_{si}$  and  $n_{si}$  are inputs of tool Treeomics. MIPUP and LICHeE require  $VAF$  values as the inputs and CITUP requires  $2 \times VAF$  as its input. To ensure all tools are evaluated on the exact same data, we use  $\frac{x_{si}}{n_{si}}$  as inputs for MIPUP and LICHeE and  $2 \times \frac{x_{si}}{n_{si}}$  as input for CITUP.

---

**Algorithm 4:** Data Simulation with a Prüfer sequence (continued)

---

**Data:**  $F_{unpack}$   
**Result:**  $n$ ;  $x$ ;  $n\_x$ ;  $\frac{x}{n}$

- 1 Let  $n$  = a empty matrix of size  $n_{sample} \times 100$ ;
- 2 Let  $x$  = a empty matrix of size  $n_{sample} \times 100$ ;
- 3 Let  $F_{unpacknoise}$  = a empty matrix of size  $n_{sample} \times 100$ ;
- 4 Let  $read\_coverage = r$ ;
- 5 **for**  $i = 0$  to  $99$  **do**
- 6     **for**  $p = 0$  to  $n_{sample} - 1$  **do**
- 7          $n[i, p]$  = a random number from a poisson distribution  $Poiss(r)$ ;
- 8          $x[i, p]$  = a random number from a binomial distribution  
 $B(n[i, p], F_{unpack}[p, i])$ ;
- 9          $\frac{x}{n}[i, p] = x[i, p]/n[i, p]$  ;
- 10     **end**
- 11 **end**
- 12  $n\_x = n - x$ ;

---

We use a simplified example with 4 clones and 10 mutations to demonstrate the algorithm described above. We mark clones from 0 to 3 and mark mutations from  $m_0$  to  $m_9$ . We generate a random sequence of length 2, say (0,1). The tree associated with it is drawn in Figure 16(1). We set a random cell population size for each clone ranging from 100 to 200. New mutations occurring in each clone are shown

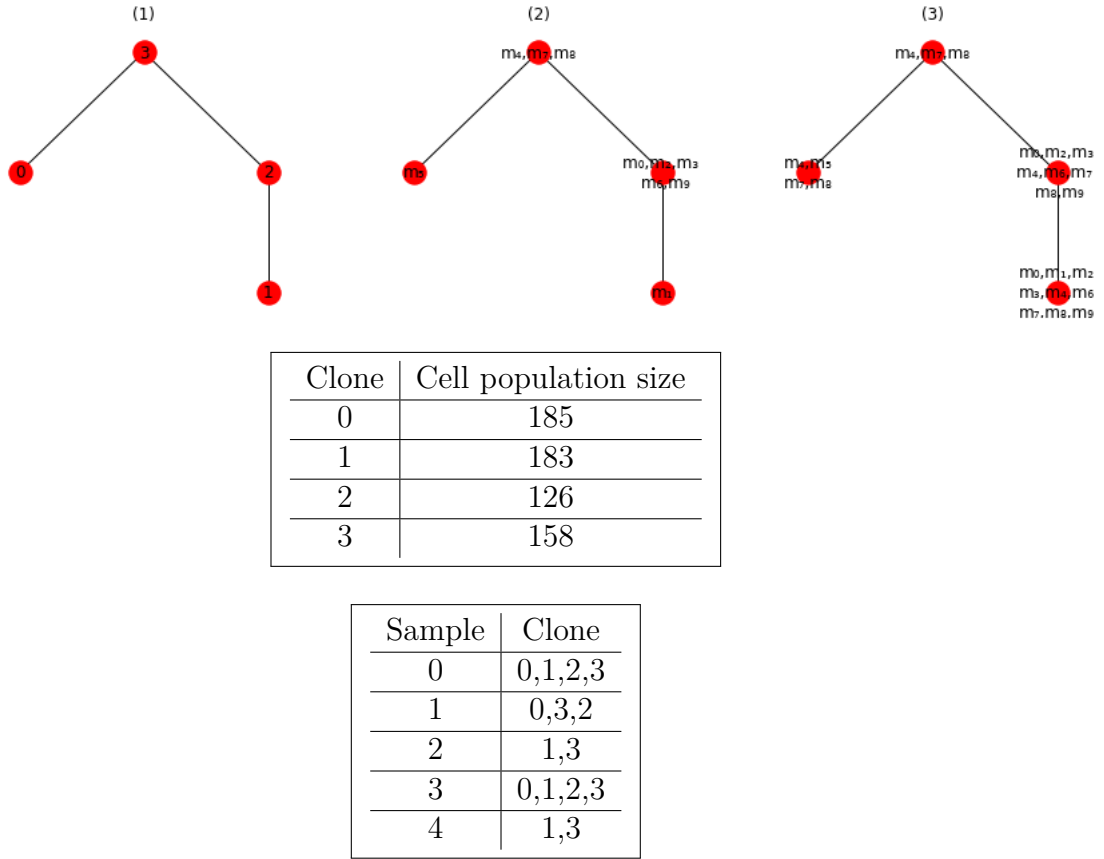


Figure 16: Simplified example with 4 clones and 10 mutations

in Figure 16(2), and mutations after inheritance are shown in Figure 16(3). We collect 5 samples from the tree. Samples are marked from 0 to 4 and clones belong to each sample are shown in Figure 14. Then matrices  $U, B, F, F_{unpack}$  (Figure 17) are generated as described above. We then did sampling with read coverage 100. Finally, matrices  $n$  and  $x$  (Figure 18) are created. Matrices  $n$  and  $x$  are inputs for tool Treomics,  $n$  and  $n - x$  are inputs for AncesTree,  $\frac{x}{n}$  is input for MIPUP and LICHeE and  $2 \times \frac{x}{n}$  is the input for CITUP.

$$U = \begin{pmatrix} 0.284 & 0.281 & 0.193 & 0.242 \\ 0.394 & 0 & 0.269 & 0.337 \\ 0 & 0.537 & 0 & 0.463 \\ 0.284 & 0.281 & 0.193 & 0.242 \\ 0 & 0.537 & 0 & 0.463 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$F = \begin{pmatrix} m_5 & m_1 & m_0, m_2, m_3, m_6, m_9 & m_4, m_7, m_8 \\ 0.142 & 0.14 & 0.237 & 0.5 \\ 0.197 & 0 & 0.134 & 0.5 \\ 0 & 0.268 & 0.268 & 0.5 \\ 0.142 & 0.14 & 0.237 & 0.5 \\ 0 & 0.268 & 0.268 & 0.5 \end{pmatrix}$$

$$F_{unpack} = \begin{pmatrix} m_0 & m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 \\ 0.237 & 0.14 & 0.237 & 0.237 & 0.5 & 0.142 & 0.237 & 0.5 & 0.5 & 0.237 \\ 0.134 & 0 & 0.134 & 0.134 & 0.5 & 0.197 & 0.134 & 0.5 & 0.5 & 0.134 \\ 0.268 & 0.268 & 0.268 & 0.268 & 0.5 & 0 & 0.268 & 0.5 & 0.5 & 0.268 \\ 0.237 & 0.14 & 0.237 & 0.237 & 0.5 & 0.142 & 0.237 & 0.5 & 0.5 & 0.237 \\ 0.268 & 0.268 & 0.268 & 0.268 & 0.5 & 0 & 0.268 & 0.5 & 0.5 & 0.268 \end{pmatrix}$$

Figure 17: Simplified example with 4 clones and 10 mutations (continued)

$$n = \begin{pmatrix} 84 & 99 & 99 & 98 & 74 & 112 & 114 & 117 & 104 & 99 \\ 104 & 101 & 107 & 105 & 85 & 116 & 102 & 90 & 111 & 126 \\ 96 & 95 & 96 & 113 & 122 & 89 & 101 & 94 & 99 & 104 \\ 109 & 102 & 103 & 103 & 88 & 105 & 104 & 84 & 100 & 99 \\ 106 & 92 & 109 & 90 & 95 & 106 & 110 & 94 & 93 & 119 \end{pmatrix}$$

$$x = \begin{pmatrix} 19 & 19 & 19 & 24 & 38 & 9 & 27 & 56 & 60 & 22 \\ 14 & 0 & 11 & 12 & 32 & 24 & 15 & 37 & 56 & 17 \\ 25 & 24 & 25 & 27 & 61 & 0 & 33 & 52 & 58 & 28 \\ 21 & 13 & 17 & 22 & 52 & 10 & 23 & 42 & 48 & 28 \\ 30 & 27 & 31 & 29 & 46 & 0 & 28 & 51 & 51 & 40 \end{pmatrix}$$

$$n - x = \begin{pmatrix} 65 & 80 & 80 & 74 & 36 & 103 & 87 & 61 & 44 & 77 \\ 90 & 101 & 96 & 93 & 53 & 92 & 87 & 53 & 55 & 109 \\ 71 & 71 & 71 & 86 & 61 & 89 & 68 & 42 & 41 & 76 \\ 88 & 89 & 86 & 81 & 36 & 95 & 81 & 42 & 52 & 71 \\ 76 & 65 & 78 & 61 & 49 & 106 & 82 & 43 & 42 & 79 \end{pmatrix}$$

$$\frac{x}{n} = \begin{pmatrix} \frac{19}{84} & \frac{19}{99} & \frac{19}{99} & \frac{24}{98} & \frac{38}{74} & \frac{9}{112} & \frac{27}{114} & \frac{56}{117} & \frac{60}{104} & \frac{22}{99} \\ \frac{14}{104} & \frac{0}{101} & \frac{11}{107} & \frac{12}{105} & \frac{32}{85} & \frac{24}{116} & \frac{15}{102} & \frac{37}{90} & \frac{56}{111} & \frac{17}{126} \\ \frac{25}{96} & \frac{24}{95} & \frac{25}{96} & \frac{27}{113} & \frac{61}{122} & \frac{0}{89} & \frac{33}{101} & \frac{52}{94} & \frac{58}{99} & \frac{28}{104} \\ \frac{21}{109} & \frac{13}{102} & \frac{17}{103} & \frac{22}{103} & \frac{52}{88} & \frac{10}{105} & \frac{23}{104} & \frac{42}{84} & \frac{48}{100} & \frac{28}{99} \\ \frac{30}{106} & \frac{27}{92} & \frac{31}{109} & \frac{29}{90} & \frac{46}{95} & \frac{0}{106} & \frac{28}{110} & \frac{51}{94} & \frac{51}{93} & \frac{40}{119} \end{pmatrix}$$

Figure 18: Simplified example with 4 clones and 10 mutations.  $n$  and  $x$  are inputs for tool Treeomics,  $n$  and  $n - x$  are inputs for Ancestree,  $\frac{x}{n}$  is input for MIPUP and LICHeE and  $2 \times \frac{x}{n}$  is the input for CITUP

Table 2: Generating probability of rooted labeled trees of (13) in Figure 10

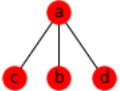






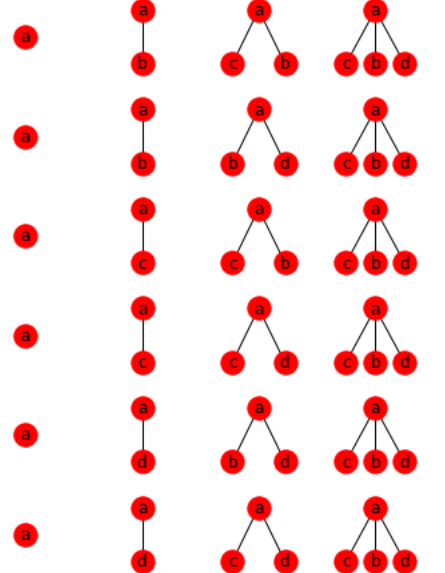
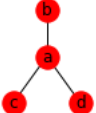


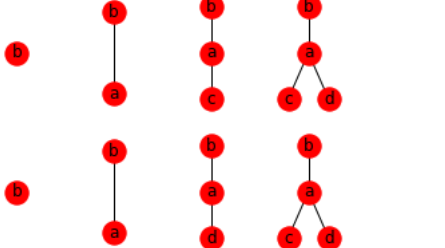
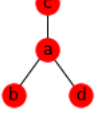
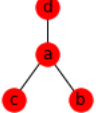
Rooted labeled trees of (13)	Generating patterns			Probability
<p>rooted labeled tree 1</p> 	<p>pattern 1</p>  <p>pattern 2</p>  <p>pattern 3</p>  <p>pattern 4</p>  <p>pattern 5</p>  <p>pattern 6</p> 		$\frac{1}{144} \times 6 = \frac{6}{144}$	
<p>rooted labeled tree 2</p> 	<p>pattern 1</p>  <p>pattern 2</p> 		$\frac{1}{144} \times 2 = \frac{2}{144}$	
<p>rooted labeled tree 3</p> 	<p>Similar with rooted labeled tree 2</p>			$\frac{2}{144}$
<p>rooted labeled tree 4</p> 	<p>Similar with rooted labeled tree 2</p>			$\frac{2}{144}$

Table 3: Generating probability of rooted labeled trees of (1) in Figure 12


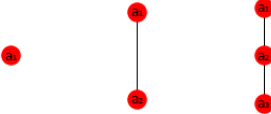
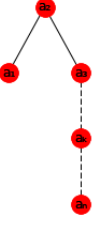
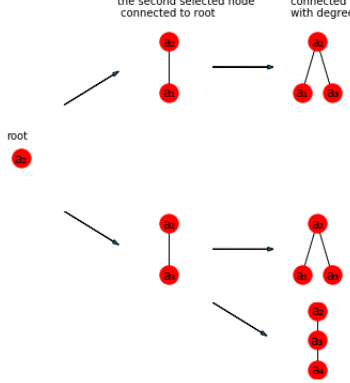
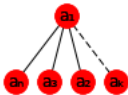
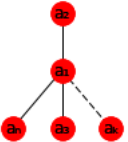
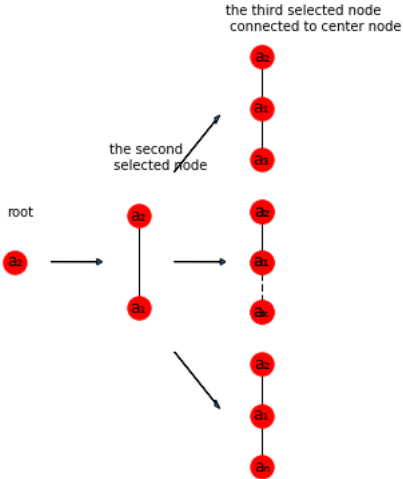
Rooted labeled trees of (1)	Generating patterns	Probability
<p>rooted labeled tree 1</p> 		$\frac{1}{n! \times (n-1)!} \times 1 = \frac{1}{n! \times (n-1)!}$
<p>rooted labeled tree 2</p> 	 <p>the second selected node connected to root</p> <p>the third selected node connected to nodes with degree one</p>	$\frac{1}{n! \times (n-1)!} \times (n-1) = \frac{1}{n! \times (n-2)!}$
Other rooted labeled trees	Omitted	Omitted



Table 4: Generating probability of rooted labeled trees of (2) in Figure 12

Rooted labeled trees of (1)	Generating patterns	Probability
<p>rooted labeled tree 1</p> 	Omitted	$\frac{1}{n! \times (n-1)!} \times (n-1)! = \frac{1}{n!}$
<p>rooted labeled tree 2</p> 		$\frac{1}{n! \times (n-1)!} \times (n-2)! = \frac{1}{n! \times (n-1)}$
Other rooted labeled trees	Omitted	Omitted

## 4 Evaluation of the Tumor Phylogenetic Reconstructing Tools with Simulated Data

### 4.1 Evaluation criteria

We set 8 evaluation criteria related to the ancestry-descendent relationship of mutation pairs. Some of our criteria are the same with the criteria of the evaluated tools, which will be explained in Subsection 4.2. Our evaluation criteria are as follows:

1. the fraction of mutation pairs which are *AD* in the true tree and are also *AD* in the predicted tree (*%AD correct*);
2. the fraction of mutation pairs which are *AD* in the true tree but are *DA* (descendent- ancestor) in the predicted tree (*%AD reversed*);
3. the fraction of mutation pairs which are *siblings* in the true tree and are also *siblings* in the predicted tree (*%Sib correct*);
4. the fraction of mutation pairs which are *AD* or *siblings* in the true tree and their relationship is reserved correctly in the predicted tree (*% (AD + Sib)*);
5. the fraction of mutation pairs which are *AD* in the true tree but are *siblings* in the predicted tree (*%AD to Sib*);
6. the fraction of mutations pairs which are *siblings* in the true tree but are *AD* in the predicted tree (*%Sib to AD*);
7. the fraction of mutations kept in the predicted tree;
8. *precision, recall, true negative rate, false positive rate, accuracy* and *F1 score* of mutation pairs with ancestor-descendant relationship:

- $precision = true\ positive / (true\ positive + false\ positive)$
- $recall\ (true\ positive\ rate) = true\ positive / (true\ positive + false\ negative)$
- $true\ negative\ rate = true\ negative / (true\ negative + false\ positive)$
- $false\ positive\ rate = false\ positive / (true\ negative + false\ positive)$
- $accuracy = (true\ positive + true\ negative) / (true\ positive + true\ negative + false\ positive + false\ negative)$
- $F1\ score = 2 / (\frac{1}{precision} + \frac{1}{recall})$

Here, *true positive (tp)* is the number of mutation pairs which are *AD* in the true tree and are also *AD* in the predicted tree, *false negative (fn)* is the number of mutation pairs which are *AD* in the true tree but are not *AD* in the predicted tree, *false positive (fp)* is the number of mutation pairs which are not *AD* in the true tree but are *AD* in the predicted tree and *true negative (tn)* is the number of mutation pairs which are not *AD* in the true tree and are also not *AD* in the predicted tree.

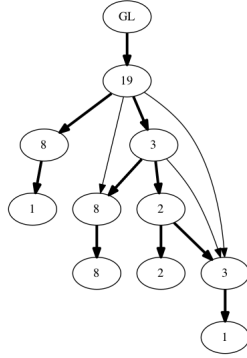


Figure 19: Evolutionary constraint network for a real dataset used in paper presenting LICHeE, The dataset contains 8 tumor samples and 55 *SSNVs*. Each node is associated with the number of *SSNVs* assigned to it. The edges represent the potential precedence relationships between the node *SSNVs*. The spanning tree reported for the real dataset is highlighted [16].

## 4.2 Overview of the evaluation criteria of the evaluated tools

Among our evaluation criteria, criteria 1 is used in papers presenting all the other evaluated tools, criteria 3, 5, 6 and 7 are used also in paper presenting LICHeE. In addition to those criteria, each paper has specific criteria related to how its tool works to reconstruct phylogenetic trees. In this subsection, we briefly introduce working procedures of each tool and then introduce the specific evaluation criteria of the paper presenting each tool.

LICHeE works by first classifying mutations into groups where each group contains mutations detected in the same subset of samples. Then the mutations in each group are further clustered according to their *VAF* similarity. After that, LICHeE constructs an evolutionary constraint network (Figure 19) to encode if a group of mutations could have preceded another group of mutations. The network is a directed acyclic graph (DAG), where each node corresponds to an *SSNV* cluster (except the root, which represents the germ line), and each edge between two nodes,  $(u \rightarrow v)$  denotes that node  $u$  could be an evolutionary predecessor of node  $v$ . In particular, an edge  $(u \rightarrow v)$  is added only if the *VAFs* of node  $u$  and  $v$  satisfy a set of constraints (in general, the *VAF* centroid vector of node  $u$  is larger than that of node  $v$ ). The constraints guarantee that the network will be acyclic. Finally, LICHeE searches spanning trees over the network to be valid phylogenetic trees. For a pair of mutations, if they occurred in two different clones and one clone was an ancestor of the other one, then these two mutations have an ancestor-descent relationship (*AD*). If they, however, occurred in two different clones and the two clones had the same parent, these two mutations have a sibling relationship (*Sib*). In all other cases, these two mutations are not related. The paper presenting LICHeE does not have specific evaluation criteria.

AncesTree works by describing the mutational process that produced a tumor by an  $n$ -clonal tree  $T$ .  $T$  is a rooted tree of  $n$  vertices for  $n$  mutations provided that each edge is labeled with exactly one mutation and no mutation appears more

than once in  $T$ . From  $T$ , AncesTree collects samples and computes matrices  $U$ ,  $B$  and  $F$  as described in our simulation algorithm. After obtaining  $VAF$  matrix  $F$ , AncesTree formalizes the problem of reconstructing the clonal evolution of a tumor as the  $VAF$  factorization problem ( $VAFFP$ ). The problem is to determine the composition of each sample, including the number and proportion of clones in each, and a tree that describes the ancestral relationships between all clones. Then, AncesTree derives a characterization of the solutions of the  $VAFFP$  as constrained spanning arborescences of a directed acyclic graph (DAG) called the ancestry graph. From this characterization, it proves that the  $VAFFP$  is NP-complete and formulates an integer linear program (ILP) to find the largest arborescence in an ancestry graph. If the largest arborescence is a spanning arborescence, a solution to the  $VAFFP$  is found. The paper presenting AncesTree includes the following specific evaluation criteria:

1. the fraction of clustered relationships between pairs of mutations that were correctly identified;
2. the fraction of incomparable relationships (i.e. neither ancestral nor clustered) between pairs of mutations that were correctly identified;
3. discrepancy between a simulated frequency matrix  $F$  and its predicted frequency matrix;
4. discrepancy between a simulated usage matrix  $U$  and its predicted usage matrix.

CITUP works by exhaustively enumerating all possible phylogenetic trees up to a fixed number of nodes, and fitting each sample into several nodes of a tree by minimizing a Bayesian information criterion on the  $VAF$  values. CITUP proposed two methods to solve the problem, in which one method guarantees an optimal solution but limit the feasible mutation size, and the second method solves the problem iteratively until convergence. For evaluation, CITUP creates a matching between the true tree and the predicted tree. The paper presenting CITUP contains the following specific evaluation criteria:

1. the fraction of correctly identified tree topologies;
2. discrepancy between clone frequency of nodes in the smallest tree and clone frequency of nodes in the largest tree;
3. the fraction of mutations which are placed in a node other than the matching node in the predicted tree;

Treomics used a Bayesian inference model to calculate the probability that a mutation is present in a sample. Then it calculates a reliability score for each possible mutation pattern (the set of samples where the variant is present) across all mutations. After that, it constructs an evolutionary conflict graph in which each node represents a mutation pattern and is assigned a weight provided by the reliability score. Two mutations are evolutionary compatible if there exists an evolutionary tree where

each mutation is only acquired once and never lost. If two nodes were evolutionarily incompatible, an edge is added between them. Finally, the phylogenetic tree is inferred from the evolutionary conflict graph. The paper presenting Treeomics has the following specific evaluation criteria:

1. whether mutations in the same clone in the true tree are assigned to the same clone in the predicted tree;

MIPUP works by first transforming a  $VAF$  value matrix into a binary one with a provided threshold value. Each row of the binary matrix represents a sample and each column represents a mutation. Then MIPUP splits each row into several rows so that the resulting matrix corresponds to a perfect phylogeny. The splitting is performed so that the resulting matrix is 'minimal'. MIPUP addresses two types of splitting problems, one is to ensure the resulting matrix has the minimum number of rows, this problem is called MinimumConflict-FreeRowSplit (MCRS). The other one is to ensure the resulting matrix has the minimum number of distinct rows, which is called MinimumDistinctConflict-FreeRowSplit (MDCRS).

### 4.3 Evaluation results

In this section, we first illustrate the predicted trees of each tool for one of our simulated data to intuitively show the performance of the tools. Then we illustrate their performance with our evaluation criteria. We varied the sample size and read coverage to assess performance of the tools under different situations. All results are averaged over 100 simulated trees to remove randomness.

#### 4.3.1 Predicted tree of the tools for one of our simulated data

In the true tree (Figure 20), the number on an edge indicates how many new mutations occurred on that edge, for example, 6 on the edge from clone 8 to clone 7 means 6 new mutations occurred there. This is also the number of new mutations in clone 7 with respect to its ancestors. In the predicted trees of MIPUP, the label on an edge provides information of the group of mutations on that edge, for example, C|11|.47  $\pm$  .1 means 11 new mutations that occurred there, and 0.47 is the mean of the  $VAF$  values in mutation group C and 0.1 is the standard deviation of those  $VAF$  values. In the predicted tree of LICHeE, the number on each node indicates the number of mutations occurred in that clone, while in the predicted tree of AncesTree, numbers in each node indicate which mutations are in that clone (mutations are marked from 0 to 99). In the predicted tree of CITUP, information of mutations is not shown on the tree but is in other output file; the number in each node is the fraction of mutations coming from each node in one of the samples. In the predicted tree of Treeomics, the number on an edge tells the number of new mutations on it.

As is seen in Figure 20, predicted trees of MIPUP and LICHeE are highly topologically similar with the true tree. The predicted tree of Treeomics also has relatively high similarity with the true tree. The predicted tree of AncesTree seems to have just kept a small amount of mutations because each node does not contain as

many mutations as in the true tree. For CITUP, there are few nodes in the predicted tree and this property might affect the performance of CITUP to some extent.

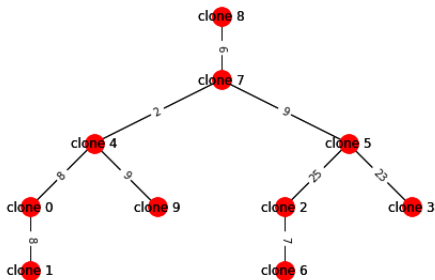
### 4.3.2 Performance of the tools on our evaluation criteria

When running MIPUP, 0.05 was used as the threshold value to transform a *VAF* value matrix into a binary one. As is illustrated in Figure 22 and 23, the performance of MIPUP is improving with increasing sample size, which is a general rule occurred for most of the other tools. The performance of MCRS problem is better than MDCRS problem. When considering different read coverages, MIPUP has its best performance with read coverage 1000, and this may indicate that a large read coverage is not necessarily result in better performance. Thus in practice, there may not be the need for large amount of reads at all times. Thus, MIPUP is an economically efficient tool. Excluding the case of 5 samples,  $\%SSNV$  is almost 1 in all the other cases (Figure 22), and this indicates MIPUP can highly preserve mutations in samples. Furthermore, the *precision*, *recall*, *true negative rate* and *accuracy* are very high and even higher than  $\%Corr Sib$ , therefore, MIPUP can predict mutation pairs of ancestor-descent relationship with high *accuracy* and it might be better at predicting *AD* than *siblings*.

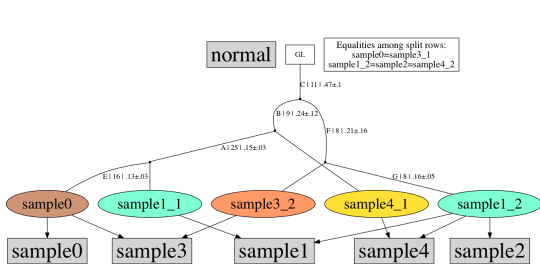
When running LICHeE, two parameters are required: Maximum *VAF* to consider an *SSNV* as robustly absent from a sample (-maxVAFAbsent) and Minimum *VAF* to consider an *SSNV* as robustly present in a sample (-minVAFPresent). Both parameters were set at 0.05 to make the running consistent with MIPUP and the other parameters as default. The results of LICHeE have the vast majority properties that the results of MIPUP have, such as increased performance with larger sample size. When considering read coverages, LICHeE seems to prefer larger read coverages than smaller ones (Figure 23 (e) and (f)). When the sample size increases, the difference between  $\%Corr AD$  and  $\%Corr Sib$  decreases (Figure 22), thus a larger sample sizes is preferable when desiring a good prediction accuracy for both *AD* and *Sibling*.

We ran AncesTree with the default values for all its parameters and ran it up to 15 samples because the running time for 20 samples is too long. AncesTree does not perform well on most of our evaluation criteria. With an increased sample size, its performance slightly decreased (Figure 22 and Figure 23 (a),(b),(c)). From  $\%SSNV$  we learned that AncesTree kept only a small fraction of mutations in its predicted tree (Figure 23 (g)), which could partly explain its poor performance. However, AncesTree has good performance on the following criteria: *precision*, *true negative rate* and *accuracy*, which are even better than those of MIPUP and LICHeE (Figure 21). Hence, although the overall performance of AncesTree is inferior, the prediction of ancestor-descent relationship among the kept mutations is highly accurate.

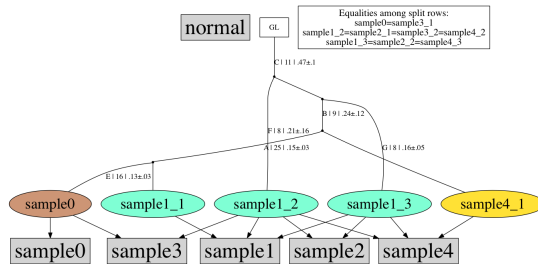
When running CITUP, the required parameter is the number of clusters in which to classify mutations. Since our simulated tree has 10 clones, this parameter was set to be 10 in order for a fair comparison. CITUP also requires to discard any mutation that is suspected or known to be homozygous, which are mutations with *VAF* values significantly larger than 0.5. From our experiments, when the read coverage is 100,



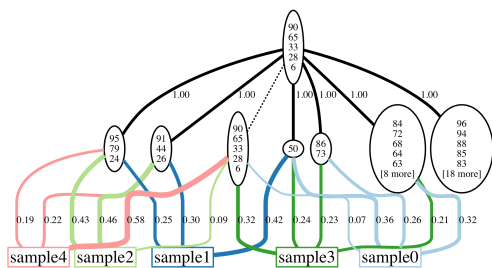
(a) True tree



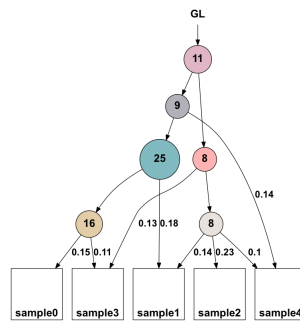
(b) Predicted tree of MIPUP (MCRS)



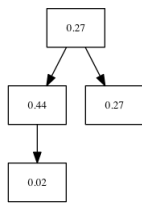
(c) Predicted tree of MIPUP (MDCRS)



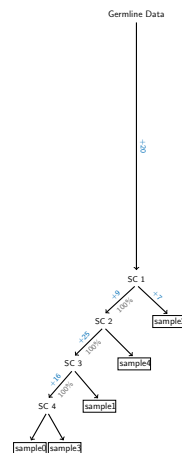
(d) Predicted tree of AncestryTree



(e) Predicted tree of LICHeE



(f) Predicted tree of CITUP



(g) Predicted tree of Treeomics

Figure 20: Predicted trees of each tool for one simulated data

the majority of the simulated datasets contain mutations with  $VAF$  values larger than 0.5 and these datasets will lead to running errors in CITUP. Thus, CITUP was run with datasets of read coverage 1000 and 10000. CITUP usually has more than one solution for one dataset and their average and best performance is illustrated with  $\%AD$  as the indicator. Some solutions have no  $AD$  in their predicted trees, such as two layer trees with no mutations in their root. When computing *precision* for such trees,  $tp$  and  $fp$  would be 0, and *precision* would be meaningless. Therefore, we discarded solutions with no  $AD$  and computed the best and average over the rest solutions. The results show, when the sample size increases from 10 to 15, the performance of  $\%Corr(AD + Sib)$  and  $\%Corr AD$  increases notably, but  $\%Sib$  decreases slightly (Figure 23 (a),(b)), which might indicate the increasing sample size is unable to increase the overall performance. When the sample size increases from 15 to 20,  $\%Corr (AD+Sib)$  and  $\%AD$  do not increase much ((Figure 23 (a),(b))), which could state that CITUP requires a certain quantity of samples, but increasing beyond that point is not necessary for increased performance. CITUP assigns no mutation in the root of its predicted tree and this can explain its comparatively low performance of  $\%Corr AD$  to some extent. Additionally, CITUP can keep all mutations in all situations (Figure 23 (c)), but its *precision* is slightly worse than other tools (Figure 23 (d)), this is in consistence with its relatively low  $\%Corr AD$ . Moreover, the average performance and the best performance do not have much difference which indicates a fair amount of similarity between solutions of a dataset.

When running Treeomics, the tool terminates with an error when sample size is 15 or 20 maybe due to some internal time limit program. Treeomics has a time limit parameter but when setting the parameter to be unlimited, the obtained solution is no longer guaranteed to be optimal. Thus, Treeomics was ran with default values for all its parameters and ran it up to 10 samples. Treeomics has relatively good performance on most criteria. The performance on  $\%Corr AD$  is better than  $\%Sib$  (Figure 23 (a),(b)) and this might point out that Treeomics is superior at predicting ancestor-descent relationships. Treeomics also has more than one solution with slight difference between average and best solution.

To summarize, MIPUP and LICHeE have the best performance: from Figure 21, they have larger area under their ROC curves and are closer to (0,1) point in the plot, compared with other tools. LICHeE has a drawback of relatively poor performance at read coverage 100 (Figure 23 (e)). This indicates MIPUP might be a less expensive tool in practice because users do not need to collect a large amount of reads. However, LICHeE seems to be slightly better at predicting  $\%Corr Sib$  than MIPUP (Figure 23 (b)). AncesTree does not perform well overall, and from the ROC curve (Figure 21), its prediction accuracy is worse than a random guess because its curve lies under the diagonal. This might be because AncesTree is unable to keep most mutations, but from the precision and-recall curve (Figure 21), its prediction is very accurate among the kept ones and even more accurate than MIPUP and LICHeE. Treeomics also has good performance, however because of its termination error, Treeomics might be suitable for data with a small sample size. CITUP also performs well although it is inferior to MIPUP and LICHeE (Figure 21), especially since it keeps all mutations. Excluding LICHeE, the other tools are not highly sensitive to read



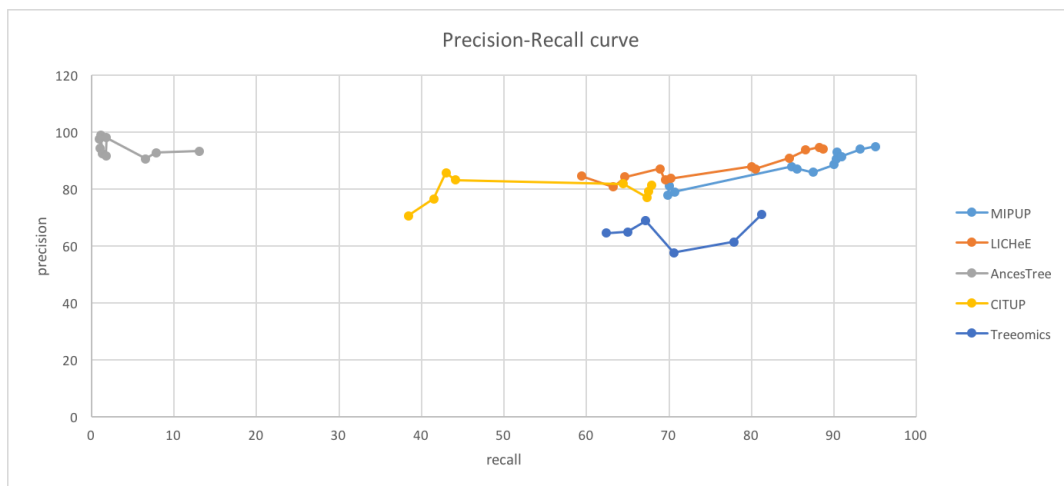
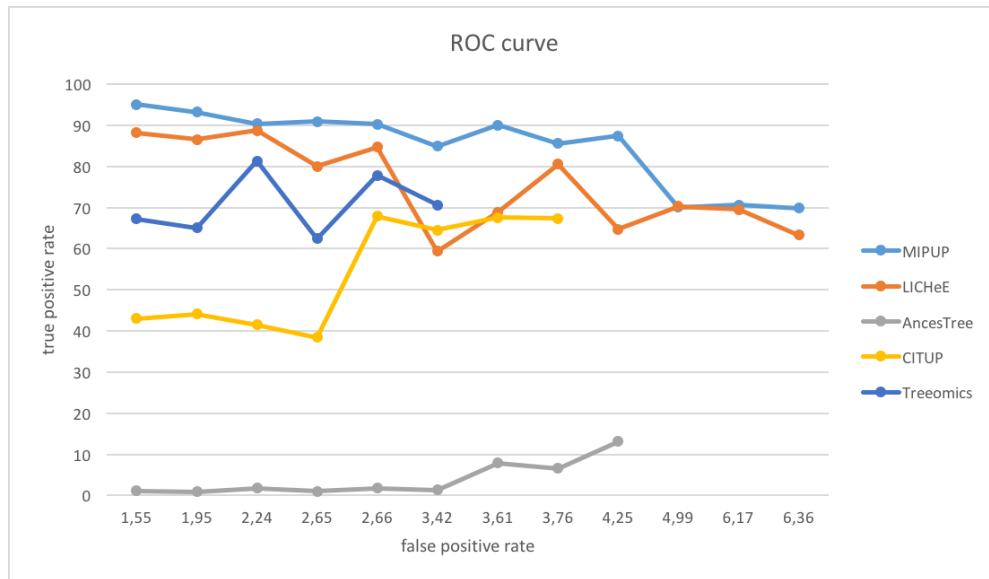


Figure 21: ROC and Precision-Recall curves

coverages (Figure 23 (e)), therefore it may not be necessary to collect a large amount of reads to increase performance. For most tools, when the sample size is larger than 10, increasing sample size does not increase performance notably (Figure 23 (a),(b)). Hence, if not demanded for extreme performance, users may collect moderate size of samples to reduce costs.

MIPUP															
number of samples	read coverage	MCRS/MDCRS	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	100	MCRS	64,44	69,82	0,08	27,96	0,1	40,94	93,37	77,8	69,82	73,59	93,64	6,36	87,54
		MDCRS	62,74	66,82	0,09	38,19	2,58	33,69	93,37	81,17	66,82	73,3	95,02	4,98	87,76
	1000	MCRS	64,4	70,06	0	29,13	0	37,51	91,49	81,03	70,06	75,15	95,01	4,99	88,72
		MDCRS	59,02	63,07	0	33,95	5,02	35,04	91,49	81,77	63,07	71,21	95,73	4,27	87,81
	10000	MCRS	64,6	70,66	0	23,29	0	46,11	93,16	79,04	70,66	74,62	93,83	6,17	87,78
		MDCRS	60,16	64,54	0	30,51	4,44	39,77	93,16	79,96	64,54	71,43	94,71	5,29	87
10	100	MCRS	79,99	84,89	0,02	48,63	0,1	31,43	98,59	87,89	84,89	86,36	96,58	3,42	93,08
		MDCRS	73,07	75,84	0,01	57,59	7,52	25,64	98,59	88,49	75,84	81,68	97,07	2,93	91,38
	1000	MCRS	81,31	87,44	0	49,66	0	35,57	99,46	85,89	87,44	86,66	95,75	4,25	93,38
		MDCRS	73,71	75,47	0	60,15	10,07	31,39	99,46	85,94	75,47	80,37	96,3	3,7	91,37
	10000	MCRS	80,84	85,55	0	52,03	0	31,15	99,15	87,06	85,55	86,3	96,24	3,76	93,04
		MDCRS	72,79	75,1	0	60,15	9,35	28,55	99,15	87,44	75,1	80,8	96,84	3,16	91,24
15	100	MCRS	84,72	90,04	0,02	56,76	0,25	28,89	99,88	88,6	90,04	89,31	96,39	3,61	94,38
		MDCRS	76,52	79,82	0,02	59,74	7,59	27,36	99,88	89,2	79,82	84,25	96,94	3,06	92,23
	1000	MCRS	87,95	90,93	0	66,8	0	24,61	100	91,38	90,93	91,15	97,35	2,65	95,41
		MDCRS	73,87	74,75	0	72,31	11,53	22,16	100	91,58	74,75	82,31	97,8	2,2	91,82
	10000	MCRS	86,25	90,26	0	68,08	0	24,68	100	90,56	90,26	90,41	97,34	2,66	95,29
		MDCRS	73,71	75,36	0	70,36	12,48	21,58	100	89,71	75,36	81,91	97,47	2,53	91,99
20	100	MCRS	86,74	90,4	0,01	55,31	0,25	22,72	99,99	93,12	90,4	91,74	97,76	2,24	95,49
		MDCRS	77,17	77,99	0,01	67,95	10,91	19,5	99,99	92,84	77,99	84,77	97,97	2,03	92,51
	1000	MCRS	92,67	95	0	78,68	0	17,54	99,94	95,01	95	95	98,45	1,55	97,32
		MDCRS	76,96	76,2	0	82	15,54	15,3	99,94	94,1	76,2	84,21	98,55	1,45	92,82
	10000	MCRS	90,54	93,2	0	73,75	0	23,87	100	93,94	93,2	93,57	98,05	1,95	96,46
		MDCRS	76,46	77,05	0	76,21	13,37	19,84	100	93,41	77,05	84,44	98,1	1,9	92,51

LICHE														
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	100	60,58	63,26	0,66	39,89	1,92	25,94	91,08	80,7	63,26	70,92	95,07	4,93	87,28
	1000	65,62	70,23	0	38,37	1,04	24,49	91,19	83,77	70,23	76,4	95,94	4,06	89,75
	10000	65,27	69,54	0,48	38,84	0,8	29,79	92,76	83,17	69,54	75,75	95,64	4,36	89,04
10	100	67,88	68,9	2,01	57,05	0,86	18,67	93	87,17	68,9	76,97	96,69	3,31	89,37
	1000	78,35	80,03	1,03	66,73	1,61	16,96	98,74	87,87	80,03	83,77	97,17	2,83	93,24
	10000	78,36	80,51	1,68	61,91	1	18,65	98,75	87,04	80,51	83,65	96,3	3,7	92,03
15	100	63,86	64,67	2,87	63,17	2,08	11,18	92,25	84,3	64,67	73,19	96,2	3,8	88,04
	1000	83,79	84,64	2,08	74,41	1,34	12,19	99,24	90,9	84,64	87,66	97,13	2,87	93,79
	10000	86	86,58	0,62	78,78	0,56	13,35	99,26	93,68	86,58	89,99	98,17	1,83	95,13
20	100	60,25	59,42	2,89	63,3	1,92	10,07	89,21	84,47	59,42	69,76	96,78	3,22	86,75
	1000	88,03	88,73	1,49	80	0,93	10,42	99,25	94,03	88,73	91,3	98,03	1,97	95,6
	10000	87,88	88,21	1,32	80,19	0,59	12,44	99,22	94,63	88,21	91,31	98,36	1,64	95,71

AnceTree														
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	100	14,52	13,11	0,02	22,49	4,97	3,76	58,09	93,38	13,11	22,99	99,74	0,26	78,42
	1000	7,53	6,54	0,07	13,54	2,93	1,42	45,82	90,68	6,54	12,2	99,79	0,21	77,45
	10000	10,04	7,83	0,01	17,65	3,28	0,9	45,43	92,78	7,83	14,44	99,82	0,18	76,83
10	100	2,86	1,8	0	9,92	0,93	0,02	31,47	98,12	1,8	3,54	99,99	0,01	74,76
	1000	3,18	1,78	0,02	9,16	0,8	0,27	30,44	91,61	1,78	3,49	99,96	0,04	77,05
	10000	2,09	1,35	0,05	6,12	1,53	0,01	26,95	92,36	1,35	2,66	99,96	0,04	75,42
15	100	2,26	1,12	0	6,98	0,39	0,02	22,38	99,06	1,12	2,21	100	0	75,33
	1000	1,54	0,95	0	3,81	0,25	0,01	17,81	97,61	0,95	1,88	100	0	75,5
	10000	1,54	1,04	0,01	4,58	0,26	0,02	18,43	94,37	1,04	2,06	99,99	0,01	75,81

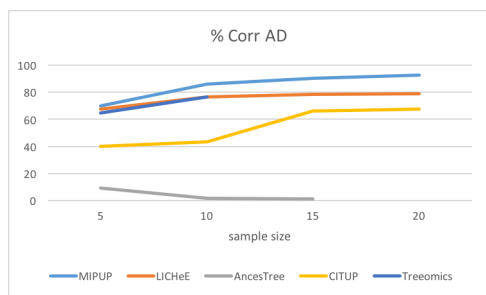
CITUP Average														
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	1000	39,02	38,42	0	37,82	3,28	33,83	100	70,55	38,42	49,75	95,2	4,8	82,08
	10000	40,64	41,47	0	40,13	6,51	30,42	100	76,61	41,47	53,81	96,21	3,79	82,72
	10	45,48	44,12	0	52,4	3,57	20,7	100	83,25	44,12	57,67	97,3	2,7	85,19
10	10000	44,03	43	0	50,56	4,65	21,94	100	85,81	43	57,29	97,93	2,07	84,58
	15	61,79	67,51	0,2	14,01	7,02	46,36	100	79,09	67,51	72,84	92,77	7,23	87,49
	10000	59,61	64,48	0	15,88	9,64	42,78	100	81,83	64,48	72,13	93,33	6,67	87,2
20	1000	61,19	67,3	0,05	10,64	9,01	45,79	100	77,1	67,3	71,87	92,36	7,64	87,43
	10000	61,52	67,92	0,05	10,52	9,04	46,02	100	81,47	67,92	74,08	93,51	6,49	88,04

CITUP Best															
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy	%Ignored
5	1000	40,04	39,9	0	36,46	2,64	35,66	100	70,99	39,9	51,09	95,06	4,94	82,32	0,51
	10000	41,76	43,71	0	38,68	5,51	32,41	100	76,73	43,71	55,69	95,99	4,01	82,97	0,87
	10	45,39	44,39	0	50,94	3,36	21,55	100	82,5	44,39	57,72	97,19	2,81	85,16	0,34
10	10000	44,33	43,33	0	50,53	4,5	21,55	100	86,48	43,33	57,73	97,97	2,03	84,7	0
	15	61,79	67,51	0,2	14,01	7,02	46,36	100	79,09	67,51	72,84	92,77	7,23	87,49	0
	10000	59,61	64,48	0	15,88	9,64	42,78	100	81,83	64,48	72,13	93,33	6,67	87,2	0
20	1000	61,19	67,3	0,05	10,64	9,01	45,79	100	77,1	67,3	71,87	92,36	7,64	87,43	0
	10000	61,52	67,92	0,05	10,52	9,04	46,02	100	81,47	67,92	74,08	93,51	6,49	88,04	0

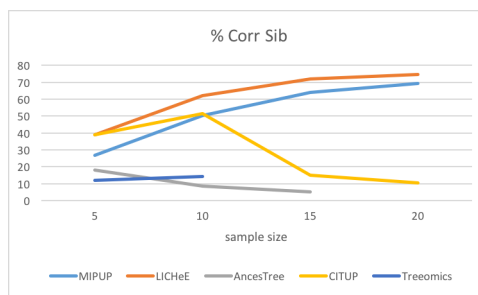
Treeomics Average														
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	100	61,05	67,2	0,03	15,28	0,04	57,77	93,37	68,87	67,2	68,02	89,66	10,34	83,74
	1000	57,8	65	0,01	11,19	0,06	53,1	91,49	64,98	65	64,99	89,42	10,58	83,19
	10000	55,84	62,39	0	9,37	0	55,79	93,16	64,52	62,39	63,44	88,47	11,53	81,57
10	100	73,32	81,28	0,01	22,51	0,03	55,83	98,6	71,1	81,28	75,85	89,3	10,7	86,61
	1000	67,84	77,85	0,05	14,92	0,05	65,7	99,46	61,38	77,85	68,64	85,38	14,62	83,03
	10000	62,33	70,59	0	5,64	0	71,81	99,15	57,63	70,59	63,46	83,41	16,59	79,45

Treeomics Best														
number of samples	read coverage	%Corr(AD+Sib)	%Corr AD	%AD reversed	%Corr Sib	%AD->Sib	%Sib->AD	%SSNVs	Precision	Recall	F1 score	True Negative Rate	False positive rate	Accuracy
5	100	62,94	69,32	0,06	16,84	0,02	57,56	93,37	69,84	69,32	69,58	89,84	10,16	84,34
	1000	59,59	67,34	0,01	9,94	0	53,77	91,49	66,08	67,34	66,7	89,5	10,5	83,81
	10000	58,41	65,55	0	8,72	0	58,81	93,16	65,71	65,55	65,63	88,51	11,49	82,34
10	100	74,52	82,92	0,01	22,31	0	55,27	98,6	71,9	82,92	77,02	89,55	10,45	87,1
	1000	70,42	80,83	0,05	15,76	0,04	66,55	99,46	62,84	80,83	70,71	85,62	14,38	83,87
	10000	67,28	76,29	0	6,02	0	74,6	99,15	59,35	76,29	66,76	83,26	16,74	80,77

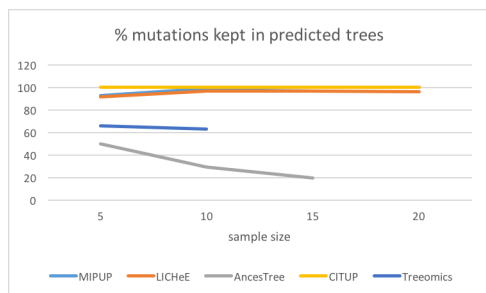
Figure 22: Performance of each tool on our evaluation criteria



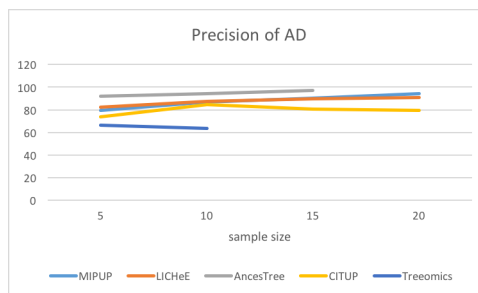
(a)



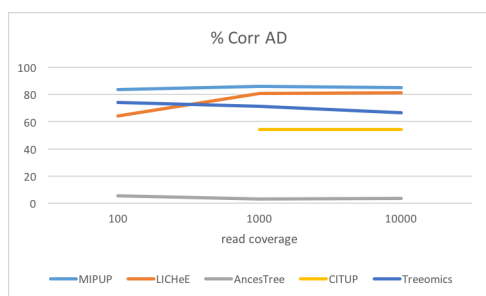
(b)



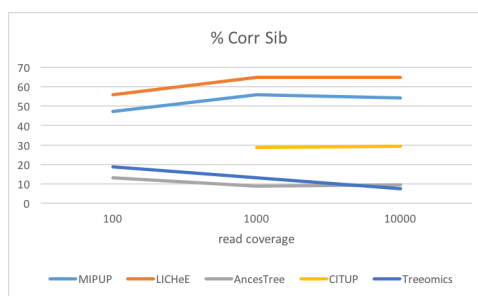
(c)



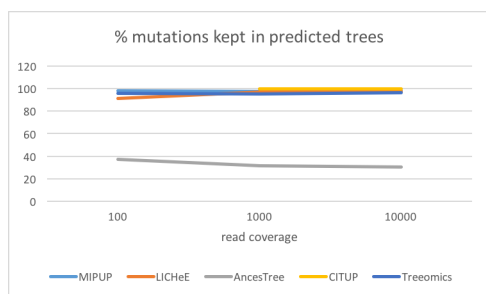
(d)



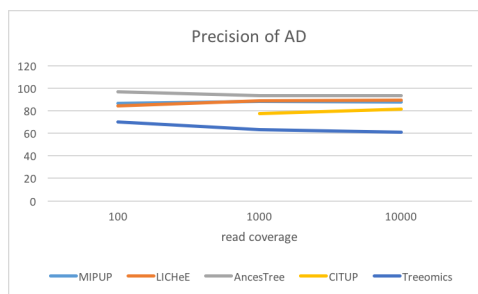
(e)



(f)



(g)



(h)

Figure 23: Plotting of the performance of each tool on part of our evaluation criteria. When considering varying sample sizes, results are average over all different read coverages; on the contrary, when considering varying read coverages, results are averaged over all different sample sizes

## 5 Discussion

There are some valuable points in both data simulation and evaluation parts which may need to be discussed further.

The largest sample size in our experiments is 20 which could be too large compared to the size of the simulated trees. In our experiments, each sample collects 2 to 4 clones, thus on average it collects 3 clones, then 20 samples may cover 60 clones of a tree. Our tree has 10 clones, which means each clone in a tree will be covered 6 times in a set of 20 samples on average. Even with 5 samples, about 15 clones can be covered which is 1.5 times of the population size. In the experiments of papers presenting the evaluated tools, most of the experiments collect up to 10 samples, only the experiments for LICHeE collects up to 20 samples perhaps because the simulated trees in the paper presenting LICHeE has hundreds to thousands of clones. Collecting a large amount of samples may be economically inefficient in practice or lead to running errors for some tools. However, since this thesis mostly focuses on evaluation, it could be beneficial to demonstrate performance of the tools on large sample sizes.

Some tools (such as AncesTree) do not perform well on our evaluation criteria, but it could be too arbitrary to conclude that they are not good tools. The reasons could be our simulated datasets are not suitable for them. For example, the paper presenting AncesTree reported that AncesTree has good performance on some real data.

Although our sample size is large, sometimes a set of samples is still unable to include all mutations of the simulated tree. Mutations not included in the samples will have VAF values equaling to 0 in each sample. It was found, however, that some tools are unable to filter out such mutations automatically so these mutations could be included in the output tree. However, this issue may not affect the overall results of the evaluated tools and the comparison among them. The reasons behind this may include the following: 1) the amount of mutations not included in samples are very rare, usually a set of samples can include more than 95% of the mutations; 2) in practice, mutations not included in the samples will not be input to tools, therefore this problem does not exist on real data.

The running time of different tools for the same dataset varied a lot. MIPUP and LICHeE are the fastest among the evaluated tools and can finish running in about 30 minutes for all situations. Treeomics is a bit slower than these two but is also fast. CITUP is relatively slow but its running time is quit stable. AncesTree is the slowest one and its running time for different datasets with the same sample size and read coverage ranges from a few minutes to dozens of hours, its long running time might partly explain its relatively small sample size (up to 6) for the experiments performed in the paper presenting AncesTree.

## 6 Summary

Studying tumor heterogeneity and inferring its phylogenies are important for developing targeted cancer therapies. High-throughput sequencing technologies allow us to detect genetic variations from biological samples. Given the variations, many computational tools have been developed to reconstruct tumor phylogenies. In this thesis, datasets consist of tumor phylogenetic trees were generated and to evaluate some recent reconstructing tools. We found relatively large differences for performance among those tools and also their strengths and shortcomings, respectively. As future work, the data simulation process could be improved to better imitate the idealized biological tumor evolutionary process. Parameters of each tool could also be studied to suit the input data better. Furthermore, it is also possible to include other evaluation criteria to catch more properties of the tools.

## References

- [1] Cancer heterogeneity: implications for targeted therapeutics. R Fisher, L Pusztai, and C Swanton
- [2] Intra-tumor heterogeneity of cancer cells and its implications for cancer treatment. Xiao-xiao Sun<sup>1</sup> and Qiang Yu<sup>1</sup>
- [3] Alberts, Bruce; Alexander Johnson; Julian Lewis; Martin Raff; Keith Roberts; Peter Walters. *Molecular Biology of the Cell*; Fourth Edition. New York and London: Garland Science. 2002. ISBN 978-0-8153-3218-3
- [4] Mccarroll, S. A.; Altshuler, D. M. (2007). "Copy-number variation and association studies of human diseases". *Nature Genetics*. 39: 37–42. PMID 17597780. doi:10.1038/ng2080.
- [5] Sharp, A. J.; Locke, D. P.; Mcgrath, S. D.; Cheng, Z; Bailey, J. A.; Vallente, R. U.; Pertz, L. M.; Clark, R. A.; Schwartz, S.; Se Graves, R. (2005). "Segmental Duplications and Copy-Number Variation in the Human Genome". *The American Journal of Human Genetics*. 77 (1): 78–88.
- [6] NCI Dictionary of Cancer Terms:  
<https://www.cancer.gov/publications/dictionaries/cancer-terms?cdrid=46586>
- [7] Nowell,P.C. (1976) The clonal evolution of tumor cell populations. *Science*, 194, 23–28.
- [8] Birbrair A, Zhang T, Wang ZM, Messi ML, Olson JD, Mintz A, Delbono O (Jul 2014). "Type-2 pericytes participate in normal and tumoral angiogenesis". *American Journal of Physiology. Cell Physiology*. 307 (1): C25–38. PMC 4080181. PMID 24788248. doi:10.1152/ajpcell.00084.2014.
- [9] prenatal assessment of genomes and exomes:  
<https://www.pageuk.org/parents/genome.html>
- [10] Paired-End Mapping Reveals Extensive Structural Variation in the Human Genome, Korb et al, *Science* 2007
- [11] Kurt Mehlhorn; Peter Sanders (2008). *Algorithms and Data topologies: The Basic Toolbox* (PDF). Springer Science Business Media. p. 52. ISBN 978-3-540-77978-0 Biggs, N. L.; Lloyd, E. K.; and Wilson, R. J. *Graph Theory* 1736-1936.
- [12] Weisstein, Eric W. "Labeled Tree." From MathWorld—A Wolfram Web Resource. <http://mathworld.wolfram.com/LabeledTree.html>

- [13] Prüfer sequence:Prüfer, H. "Neuer Beweis eines Satzes über Permutationen." Arch. Math. Phys. 27, 742-744, 1918. Skiena, S. Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica. Reading, MA: Addison-Wesley, 1990.
- [14] Kajimoto, H. (2003). "An Extension of the Prüfer Code and Assembly of Connected Graphs from Their Blocks". Graphs and Combinatorics. 19: 231–239.
- [15] A. Hujdurović, E. Husić, M. Mehine, M. Milanič, R. Rizzi and A.I. Tomescu (2017). MIPUP: Minimum perfect unmixed phylogenies for multi-sampled tumors via branchings in graphs and ILP
- [16] Popic, V. et al. (2015). Fast and scalable inference of multi-sample cancer lineages. Genome Biology, 16(1), 1–17
- [17] M. El-Kebir, L. Oesper, H. Acheson-Field, and B.J. Raphael. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data Bioinformatics
- [18] Malikic, S. et al. (2015). Clonality inference in multiple tumor samples using phylogeny. Bioinformatics, 31(9), 1349–1356.
- [19] Johannes G. Reiter.Reconstructing phylogenies of metastatic cancers
- [20] Birbrair A, Zhang T, Wang ZM, Messi ML, Olson JD, Mintz A, Delbono O (Jul 2014). "Type-2 pericytes participate in normal and tumoral angiogenesis". American Journal of Physiology. Cell Physiology. 307 (1): C25–38. PMC 4080181Freely accessible. PMID 24788248. doi:10.1152/ajpcell.00084.2014.
- [21] "II Neoplasms". World Health Organization. Retrieved 19 June 2014.
- [22] Abrams, Gerald. "Neoplasia I". Retrieved 23 January 2012.
- [23] "Cancer - Activity 1 - Glossary, page 4 of 5". Retrieved 2008-01-08.
- [24] Handa O, Naito Y, Yoshikawa T (2011). "Redox biology and gastric carcinogenesis: the role of Helicobacter pylori". Redox Rep. 16 (1): 1–7. PMID 21605492. doi:10.1179/174329211X12968219310756
- [25] Halford S, Rowan A, Sawyer E, Talbot I, Tomlinson I (June 2005). "O(6)-methylguanine methyltransferase in colorectal cancers: detection of mutations, loss of expression, and weak association with G:C>A:T transitions". Gut. 54 (6): 797–802. PMC 1774551Freely accessible. PMID 15888787. doi:10.1136/gut.2004.059535.
- [26] Lee KH, Lee JS, Nam JH, Choi C, Lee MC, Park CS, Juhng SW, Lee JH (October 2011). "Promoter methylation status of hMLH1, hMSH2, and MGMT genes in colorectal cancer associated with adenoma-carcinoma sequence". Langenbecks Arch Surg. 396 (7): 1017–26. PMID 21706233. doi:10.1007/s00423-011-0812-9

- [27] Neoplasm Cooper GM (1992). Elements of human cancer. Boston: Jones and Bartlett Publishers. p. 16. ISBN 978-0-86720-191-8.
- [28] Truninger K, Menigatti M, Luz J, Russell A, Haider R, Gebbers JO, Bannwart F, Yurtsever H, Neuweiler J, Riehle HM, Cattaruzza MS, Heinemann K, Schär P, Jiricny J, Marra G (May 2005). "Immunohistochemical analysis reveals high frequency of PMS2 defects in colorectal cancer". *Gastroenterology*. 128 (5): 1160–71. PMID 15887099. doi:10.1053/j.gastro.2005.01.056
- [29] Navigating the Challenge of Tumor Heterogeneity in Cancer Therapy Clare Fedele, Richard W. Tothill and Grant A. McArthur DOI: 10.1158/2159-8290.CD-13-1042 Published February 2014
- [30] Tumour heterogeneity and the evolution of polyclonal drug resistance react-empty: 83 react-empty: 84 react-empty: 85 react-empty: 86 react-empty: 87 react-empty: 88 react-empty: 89 Author links open overlay panel Rebecca A.BurrellabCharlesSwantonab
- [31] DNA Molecular Biology of the Cell. 4th edition. Alberts B, Johnson A, Lewis J, et al.