Sapling: Inferring and Summarizing Tumor Phylogenies from Bulk Data Using Backbone Trees

Yuanyuan Qi
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Mohammed El-Kebir[1]
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
Cancer Center at Illinois, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Abstract

Cancer phylogenies are key to understanding tumor evolution. There exist many important downstream analyses that take as input a single or a small number of trees. However, due to uncertainty, one typically infers many, equally-plausible phylogenies from bulk DNA sequencing data of tumors. We introduce Sapling, a heuristic method to solve the Backbone Tree Inference from Reads problem, which seeks a small set of backbone trees on a smaller subset of mutations that collectively summarize the entire solution space. Sapling also includes a greedy algorithm to solve the Backbone Tree Expansion from Reads problem, which aims to expand an inferred backbone tree into a full tree. We prove that both problems are NP-hard. On simulated and real data, we demonstrate that Sapling is capable of inferring high-quality backbone trees that adequately summarize the solution space and that can be expanded into full trees.

2012 ACM Subject Classification Applied computing → Computational biology

Keywords and phrases Cancer, intra-tumor heterogeneity, consensus, maximum agreement

Digital Object Identifier 10.4230/LIPIcs.WABI.2024.7

Supplementary Material *Software (Source code)*: `https://github.com/elkebir-group/Sapling` archived at `swh:1:dir:5b1f3fb5f04eaee56b52b596583bd6826a815230`

Funding *Mohammed El-Kebir*: National Science Foundation award number CCF 2046488.

1 Introduction

Cancer results from an evolutionary process during which somatic mutations accumulate in a population of cells [23]. This process results in intra-tumor heterogeneity, i.e. the presence of multiple clones with distinct sets of mutations, with important implications on cancer treatment [20]. Researchers model cancer evolution with a *phylogeny*, which is a rooted tree whose nodes correspond to clones. These trees are used in several downstream analyses [26]. These downstream analyses typically require a single or a small number of phylogenies per patient. However, deconvolution of bulk DNA measurements may lead one to infer a large solution space of equally-plausible phylogenies [25].

There are three classes of methods that attempt to overcome this mismatch between the existence of large solution spaces and downstream analysis requirements. First, several approaches attempt to sample a small number of high-likelihood trees [5, 18, 19, 29]. Second, there exist several methods that attempt to summarize a given solution space of trees with one or more consensus trees [1, 6, 9–12, 16]. Another approach that also belongs to this class is SubMARine, which, rather than inferring a complete tree, returns a directed acyclic graph

[1] To whom correspondence should be addressed

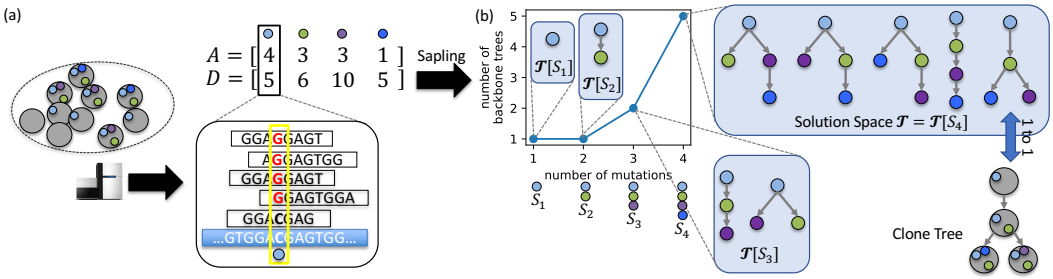

Figure 1 Overview of Sapling. (a) Bulk DNA sequencing, alignment and SNV calling results in matrices A and D of variant and total read counts of n SNVs in m samples. (b) Sapling is a heuristic for the Backbone Tree Inference from Reads problem, returning a small set of backbone trees for a given number ℓ of mutations. Here, with $\ell = 3$ mutations, the solution space $\mathcal{T}$ of 5 mutation trees can be summarized with two backbone trees $\mathcal{T}[S_3]$.

indicating ancestral relationships in the solution space [27]. Third, there exist approaches that use repeated evolutionary trajectories inferred from patient cohorts to reduce the number of solutions per patient [2, 4, 13, 14, 17].

These three classes of methods come with their own limitations. The sampling methods, which are typically MCMC-based, exhibit great bias to certain solutions [25], and thus may not infer a representative set of solutions. The consensus methods, including methods that utilize repeated evolutionary trajectories, require an exhaustive enumeration of all plausible trees, which is impractical to obtain when the set of possible trees is large.

To overcome these limitations, we introduce Sapling, a method that given read count data infers a small set of *backbone trees* on a smaller subset of mutations that collectively summarize the solution space (Fig. 1). We note that backbone trees are similar to the concept of a maximum-agreement subtree (MAST) in species phylogenetics [28], with a key distinction being that tumor phylogenies are node-labeled trees whereas species phylogenies are leaf-labeled trees. Using simulations, we show that the backbone trees returned by Sapling provide a good summary of the possible trees and can be expanded into full trees that are of higher quality than current state-of-the-art tree inference methods [18, 19, 29]. Finally, we demonstrate how Sapling can be applied to comprehensively summarize non-small lung cancer solution spaces with a small number of backbone trees [15].

2 Problem Statement

Due to uncertainty, cancer phylogeny inference algorithms typically infer a set $\mathcal{T}$ of mutation trees from bulk sequencing data rather than a single tree. In this work, we consider trees inferred under the infinite sites assumption (ISA), meaning that each mutation is gained exactly once and never subsequently lost. We note that while this assumption does not generally hold particularly due to copy-number loss, tumor phylogeny pipelines include mutation clustering correcting for such events, yielding clusters of mutations that adhere to the ISA. Under the ISA, the solution space $\mathcal{T}$ consists of rooted trees T whose nodes $V(T)$ are labeled by mutations $[n] = \{1, \ldots, n\}$ – in practice, we will view mutation clusters as individual mutations. As such, we refer to nodes and mutations interchangeably. We write $u \preceq_T v$ if node or mutation u occurs on the unique path from the root $r(T)$ to node v – note that $\preceq_T$ is reflexive, i.e., it holds that $u \preceq_T u$ for mutations u. We denote the set of children of a node v of tree T by $\delta_T(v)$. We denote the parent of a node $v \neq r(T)$ by $\pi_T(v)$. Our

goal is to identify common features or backbone trees on a smaller set $S \subseteq [n]$ of mutations that best characterize the diversity of the solution space $\mathcal{T}$. To that end, we define backbone trees as follows.

▶ **Definition 1.** *A rooted tree $T[S]$ is a* backbone tree *of a tree T on mutations $S \subseteq V(T)$ provided $u \preceq_T v$ if and only if $u \preceq_{T[S]} v$ for all mutations $u, v \in S$.*

Mathematically, T is a subdivision or expansion of $T[S]$ such that the backbone tree $T[S]$ is obtained from T by contracting nodes $V(T) \setminus S$. Rather than considering a single backbone tree on a subset S of mutations, we wish to identify a backbone tree set $\mathcal{T}[S]$ that collectively forms backbone trees of all trees $\mathcal{T}$ on the full mutation set $[n]$.

▶ **Definition 2.** *Given a set $\mathcal{T}$ of trees on n mutations, the corresponding* backbone tree set *$\mathcal{T}[S]$ for a subset $S \subseteq [n]$ of mutations consists of all backbone trees $T[S]$ of all trees $T \in \mathcal{T}$.*

Importantly, $|\mathcal{T}[S]| \leq |\mathcal{T}|$ for all $S \subseteq [n]$. The key question is which subset $S \subseteq [n]$ of mutations provides an accurate summary of $\mathcal{T}$? Ideally, we wish to simultaneously include as many mutations as possible in S, i.e. maximize $|S|$, while also minimizing the number of backbone trees $|\mathcal{T}[S]|$. However, there is a tradeoff between both criteria. One can set $S = [n]$, thus maximizing $|S|$, but this would lead to as many backbone trees as there are input trees, i.e. $\mathcal{T}[S] = \mathcal{T}$, which does not provide a summary of $\mathcal{T}$. On the other hand, setting S to contain no mutations or just a single mutation would lead to a backbone tree set consisting of a single backbone tree composed of at most one mutation; thus while minimizing $|\mathcal{T}[S]| = 1$, this does not provide any useful information that is particular to $\mathcal{T}$. To model this tradeoff, we formulate the following two problem statements, constraining either the number $|S|$ of mutations is constrained or the number $|\mathcal{T}[S]|$ of backbone trees.

▶ **Problem 1** (Minimum Cardinality Backbone Trees)**.** *Given a set $\mathcal{T}$ of trees on n mutations and parameter $\ell \in [n]$, find a subset $S \subseteq [n]$ of ℓ mutations and corresponding backbone tree set $\mathcal{T}[S]$ such that $\mathcal{T}[S]$ has minimum cardinality among all backbone tree sets induced by ℓ mutations.*

▶ **Problem 2** (Maximum Mutation Backbone Trees)**.** *Given a set $\mathcal{T}$ of trees on n mutations and parameter $\tau \in \mathbb{N}$, find a maximum-cardinality subset $S \subseteq [n]$ of mutations and corresponding backbone tree set $\mathcal{T}[S]$ such that $|\mathcal{T}[S]| \leq \tau$.*

A special version of this problem arises when $\tau = 1$. In that case, we are seeking a maximum-cardinality set S of mutations and a single corresponding backbone tree $T[S]$ on which all trees in $\mathcal{T}$ agree. In our final problem, we seek to expand a given backbone tree into a full tree.

▶ **Problem 3** (Backbone Tree Expansion)**.** *Given trees $\mathcal{T}$ on n mutations and a tree $T[S]$ on a subset $S \subseteq [n]$ of mutations, find a tree $T \in \mathcal{T}$ such that $T[S]$ is a backbone tree of T.*

Backbone Tree Inference and Expansion from Reads

In practice, we are not given the set $\mathcal{T}$ of phylogenetic trees. While there exist many methods for inferring such a set via sampling [5, 18, 19, 29] or enumeration [8, 21], obtaining the complete set $\mathcal{T}$ of trees might be infeasible due to its sheer size [25]. Therefore, we propose to infer backbone trees directly from read count data obtained from bulk DNA sequencing of m samples (regional or temporal) from the same tumor. More specifically, we are given

two matrices $A, D \in \mathbb{N}^{m \times n}$ where $A = [a_{p,i}]$ indicates the number of reads supporting the variant allele and $D = [d_{p,i}]$ indicates the total number of reads at each mutation locus in each sample.

To pose the problem, we are interested in computing the probability $\Pr(T \mid A, D)$ of a tree T given the data A, D. To compute this probability, we note that the number $a_{p,i}$ of variant read counts at mutation locus i in sample p depends on the total number $d_{p,i}$ of reads, i.e. $0 \leq a_{p,i} \leq d_{p,i}$, and the (latent) frequency $f_{p,i} \in [0, 1]$ of mutation i in sample p. Due to bulk DNA sequencing, each sample p is a mixture of different tumor clones such that each clone either contains or does not contain mutation i; therefore the frequency $f_{p,i}$ ranges between 0 and 1 rather than being either 0 or 1. Typically, one models variant read counts $A = [a_{p,i}]$ as binomial distributions, i.e. $a_{p,i} \sim \text{binom}(d_{p,i}, f_{p,i})$. Letting $F = [f_{p,i}]$ be the $m \times n$ frequency matrix, and using the independence of mutations and samples, we thus have

$$\Pr(A \mid D, F) = \prod_{p=1}^{m} \prod_{i=1}^{n} \binom{d_{p,i}}{a_{p,i}} (f_{p,i})^{a_{p,i}} (1 - f_{p,i})^{d_{p,i} - a_{p,i}}. \tag{1}$$

As discussed in [7], frequencies F depend on a tree T. That is, a tree T under the ISA constrains frequencies $F = [f_{p,i}]$ as

$$f_{p,i} \geq \sum_{j \in \delta_T(i)} f_{p,j} \qquad \forall p \in [m], i \in [n]. \tag{SC}$$

This is also known as the sum condition (SC). To compute the desired probability $\Pr(T \mid A, D)$, we apply Bayes' rule, yielding $\Pr(T \mid A, D) = [\Pr(A, D \mid T) \Pr(T)] / \Pr(A, D)$. Since we observe A, D, we have that $\Pr(A, D)$ is constant. Moreover, using a flat prior on $\Pr(T)$, we obtain $\Pr(T \mid A, D) \propto \Pr(A, D \mid T)$. We now have

$$\begin{aligned} \Pr(A, D \mid T) &= \int_F \Pr(A \mid D, F) \Pr(F \mid T)\, dF && (2) \\ &\propto \int_F \Pr(A \mid D, F) \cdot \mathbf{1}\{F, T \text{ satisfy (SC)}\}\, dF && (3) \\ &\geq \max_F \Pr(A \mid D, F) \cdot \mathbf{1}\{F, T \text{ satisfy (SC)}\}. && (4) \end{aligned}$$

In other words, we approximate the probability $\Pr(A, D \mid T)$ of read counts A, D given a tree T by seeking a frequency matrix F such that F and T satisfy (SC) and $\Pr(A \mid D, F)$ is maximum. This is equivalent to solving the following optimization problem.

▶ **Definition 3.** *The* log-likelihood $\mathcal{L}(A, D \mid T)$ *of a rooted tree* T *and read counts* A, D *equals* $\max_F \mathcal{L}(A, D \mid F)$ *s.t.* (SC) *where* $\mathcal{L}(A, D \mid F)$ *is defined as* $\sum_{p=1}^{m} \sum_{i=1}^{n} \left[a_{p,i} \log f_{p,i} + (d_{p,i} - a_{p,i}) \log (1 - f_{p,i}) \right]$.

Computing $\mathcal{L}(A, D \mid T)$, or equivalently $-\mathcal{L}(A, D \mid T)$, requires solving a convex optimization problem subject to linear constraints, which can be solved in polynomial time (for a fixed error tolerance) using interior point methods [22]. To allow one to obtain near-maximum likelihood solutions, the user may specify the parameter $\rho \in [0, 1]$ yielding the solution space $\mathcal{T}^{(\rho)}$ of trees that are most a factor of ρ removed from maximum likelihood, formally defined as follows.

▶ **Definition 4.** *Given* $\rho \in [0, 1]$ *and read counts* $A, D \in \mathbb{N}^{m \times n}$*, the set* $\mathcal{T}^{(\rho)}$ *includes all trees* T *such that* $\Pr(A, D \mid T) \geq \rho \Pr(A, D \mid T^*)$ *where* T^* *is a tree on* n *mutations that maximizes* $\Pr(A, D \mid T^*)$.

Thus, we have $\mathcal{T}^{(\rho_1)} \subseteq \mathcal{T}^{(\rho_2)}$ for all $0 \leq \rho_1 \leq \rho_2 \leq 1$. Specifically, for $\rho = 0$ the set $\mathcal{T}^{(0)}$ contains all n^{n-1} rooted trees on n mutations [3]. This leads to the following updated problem statements.

▶ **Problem 4** (Minimum Cardinality Backbone Trees from Reads)**.** *Given variant and total read counts $A, D \in \mathbb{N}^{m \times n}$ for n mutations in m samples and parameters $\ell \in [n]$ and $\rho \in [0, 1]$, find a subset S of ℓ mutations and corresponding backbone tree set $\mathcal{T}^{(\rho)}[S]$ such that $\mathcal{T}^{(\rho)}[S]$ has minimum cardinality among all backbone tree sets induced by ℓ mutations on trees $\mathcal{T}^{(\rho)}$.*

▶ **Problem 5** (Maximum Mutation Backbone Trees from Reads)**.** *Given variant and total read counts $A, D \in \mathbb{N}^{m \times n}$ for n mutations in m samples and parameters $\tau \in \mathbb{N}$ and $\rho \in [0, 1]$, find a maximum-cardinality subset $S \subseteq [n]$ of mutations and backbone tree set $\mathcal{T}^{(\rho)}[S]$ such that $|\mathcal{T}^{(\rho)}[S]| \leq \tau$.*

▶ **Problem 6** (Backbone Tree Expansion from Reads)**.** *Given variant and total read counts $A, D \in \mathbb{N}^{m \times n}$ for n mutations in m samples and a tree $T[S]$ on a subset $S \subseteq [n]$ of mutations, find a tree T^* such that (i) T^* is a tree on n mutations that maximizes $\Pr(A, D \mid T^*)$ and (ii) $T[S]$ is a backbone tree of T^*.*

3 Methods

In this section, we introduce the two algorithms that make up Sapling. First, we introduce a heuristic to solve the two Backbone Trees from Reads problems subject to either a constraint on the number of mutations or the number of backbone trees. Second, we introduce a heuristic to solve the Backbone Tree Expansion from Reads problem.

3.1 Enumerating backbone trees

Given $A = [a_{p,i}]$ and $D = [d_{p,i}]$ it is clear that $\hat{F} = [\hat{f}_{p,i}]$ where $\hat{f}_{p,i} = a_{p,i}/d_{p,i}$ is the frequency matrix that maximizes $\mathcal{L}(A, D \mid T)$ when ignoring the sum condition (SC). The question whether there exists a tree T satisfying (SC) for a given frequency matrix was shown to be NP-complete when F contains $m \geq 2$ samples [8]. This means that the two Backbone Trees from Reads problem are NP-hard when $m \geq 2$. To see why, we can set $\rho = 1$ and solve the two problems with either $\ell = n$ or $\tau = n^{n-1}$. This will return one or more trees on all n mutations. If the likelihood $\mathcal{L}(A, D \mid T)$ of any such tree T equals the likelihood $\mathcal{L}(A, D \mid \hat{F})$ then there exists a tree that satisfies the sum condition with $\hat{F}$, leading to the following hardness result.

▶ **Theorem 5.** *The two Backbone Trees from Reads problems are NP-hard even when $m = 2$.*

Proof. We show hardness by a polynomial-time reduction from the Perfect Phylogeny Mixture Deconvolution (PPMD) problem [7, 8, 25] of deciding whether there exists a tree T satisfying (SC) for a given frequency matrix $F \in [0, 1]^{m \times n}$. This decision problem was shown to NP-complete when F contains $m \geq 2$ samples [8]. Without loss of generality, we may assume that F is rational (the reduction from Subset Sum presented in [8] works for rational values). Thus, we have $f_{p,i} = a_{p,i}/d_{p,i}$ where $a_{p,i}, d_{p,i} \in \mathbb{N}$ and $a_{p,i} \leq d_{p,i}$ for each sample $p \in [m]$ and mutation $i \in [n]$. These entries correspond to variant read count matrix $A = [a_{p,i}]$ and total read count matrix $D = [d_{p,i}]$. This reduction takes polynomial time.

We claim that Problem 4 with read counts A, D obtained from F is NP-hard when $m \geq 2$, $\rho = 1$ and $\ell = n$. Moreover, we claim that Problem 5 with read counts A, D obtained from F is NP-hard when $m \geq 2$, $\rho = 1$ and $\tau = n^{n-1}$. Solving either problem will return a set $\mathcal{T}$ of trees. Let $T \in \mathcal{T}$ be one such solution tree. Clearly, T is a tree on n mutations due to the constraint $\ell = n$ and $\tau = n^{n-1}$. Since F is the maximum likelihood estimator of the binomial proportions of A, D, we have that $\mathcal{L}(A, D \mid T) \leq \mathcal{L}(A, D \mid F)$. Furthermore, the likelihood $\mathcal{L}(A, D \mid T)$ equals the likelihood $\mathcal{L}(A, D \mid F)$ if and only if T satisfies (SC). That is, we can verify whether this bound is tight by simply checking whether F, T satisfy (SC), which takes polynomial time. ◀

We note that the Maximum Mutation Backbone Trees from Reads problem, where we are given an upper bound τ of backbone trees, can be solved by repeatedly solving the (Minimum Cardinality Backbone Trees from Reads) problem, where the number ℓ of mutations is fixed. That is, starting with $\ell = 1$, we obtain the backbone tree set $\mathcal{T}_\ell$ and increment ℓ until the resulting number $|\mathcal{T}_\ell|$ of trees is greater than τ. We then return $|\mathcal{T}_{(\ell-1)}|$. Since the two Backbone Trees from Reads problems are hard, we introduce the following heuristic.

3.1.1 A naive approach

In our first approach, we propose to build the backbone trees iteratively. We initialize the set $\mathcal{T}_1$ with a single tree T containing a single mutation (can be any mutation). Then at each iteration $k \geq 1$, given the current set $\mathcal{T}_k$ on the same subset S_k of mutations, we extend each tree $T \in \mathcal{T}_k$ by adding a new mutation $i \in [n] \setminus V(T)$ at each possible location. Specifically, we may either extend T by adding the edge $(i, r(T))$, or for an existing node $j \in V(T)$ we insert the new edge (j, i) and distribute the original children $\delta_T(j)$ among nodes i and j. This results in a new set $\mathcal{T}_{k+1}$ of trees on mutations $S_k \cup \{i\}$.

In order to assess whether an expanded tree $T' \in \mathcal{T}_{k+1}$ is a good backbone tree on the mutation set $S_{k+1} = S_k \cup \{i\}$, we evaluate the likelihood $\mathcal{L}(A[S_{k+1}], D[S_{k+1}] \mid T')$. That is, we take the submatrices $A[S_{k+1}]$ of A and $D[S_{k+1}]$ of D with columns corresponding to mutations S_{k+1}. Computing this likelihood requires solving a convex optimization problem and, as mentioned before, this can be solved efficiently in polynomial time with a constant error tolerance. We calculate $\mathcal{L}(A[S_{k+1}], D[S_{k+1}] \mid T')$ for all trees T' in $\mathcal{T}_{k+1}$, retaining only those trees that have a likelihood of at least $\rho \cdot \Pr(A[S_{k+1}], D[S_{k+1}] \mid T^*)$, where T^* is the maximum likelihood tree among $\mathcal{T}_{k+1}$. We terminate after iteration $\ell - 1$, returning the set S_ℓ of mutations and backbone tree set $\mathcal{T}_\ell$.

However, this algorithm does not work well in practice for two key reasons. First, the order in which mutations are considered does affect the number of resulting backbone trees. One might overcome this limitation by exploring different permutations of mutations, but this becomes quickly intractable as there are $n!$ permutations. Ideally, one would be able to determine a good permutation of mutations ahead of time, and only consider this single permutation when enumerating. Second, note there are $2^{|\delta_T(j)|} = O(2^n)$ possible expanded trees T' when expanding a single mutation j of a partial tree T. This would make it impossible to explore the entire set of possible backbone trees at each iteration. Therefore, we need additional criteria to prune the search space.

3.1.2 Adding mutations ordered by $\hat{F}$

While we do not know the true latent frequencies of the mutations, $\hat{F} = [\hat{f}_{p,i}]$ where $\hat{f}_{p,i} = a_{p,i}/d_{p,i}$ can serve as a good estimator of the latent frequencies. As such, we propose to sort the n mutations in descending order based on $\sum_{p=1}^{m} \hat{f}_{p,i}$ and consider the mutations

according to this order when enumerating backbone trees. Intuitively, this order will start by adding mutations that are closest to the root and leave the mutations that are farthest away as the last mutations to be added. We note that Orchard uses this same ordering when sampling complete trees from the solution space [19].

3.1.3 Pruning the search space

To avoid considering $O(2^n)$ possible trees when expanding a partial tree T, we propose to place a new mutation i either (i) as the new root of T, or (ii) as a new leaf of T, or (iii) split an existing edge $(\pi_T(j), j)$ of T inserting edges $(\pi_T(j), i)$ and (i, j). Importantly, there are only $O(n)$ such possible expansions of a given tree T. There is a theoretical justification for case (ii) of this pruning step when the n mutations are sorted such that $\sum_{p=1}^{m} \hat{f}_{p,i} \geq \sum_{p=1}^{m} \hat{f}_{p,j}$ for any $1 \leq i < j \leq n$. If the sequencing depth is large enough, $\hat{F}$ accurately reflects the latent frequencies of the mutations – i.e. $\hat{F} = [\hat{f}_{p,i}]$ is an unbiased maximum likelihood estimator if the variants read count are indeed binomially distributed. In this case, for any $\rho > 0$, each tree $T \in \mathcal{T}^{(\rho)}$ must adhere to (SC) with respect to $\hat{F}$. As such, we have that a new mutation $j > i$ cannot be a parent of a previous mutation i.

▶ **Theorem 6.** *Let $0 < \rho \leq 1$ and let $\mathcal{T}^{(\rho)}$ be the set of corresponding trees on n mutations. As $d_{p,i} \to \infty$ for all p and i, then if $\sum_{p=1}^{m} \hat{f}_{p,i} > \sum_{p=1}^{m} \hat{f}_{p,j}$, it holds that $j \not\prec_T i$ for all trees $T \in \mathcal{T}^{(\rho)}$.*

Proof. We prove this by contradiction, assuming there exists a tree $T \in \mathcal{T}^{(\rho)}$ with mutations i, j such that mutation j is ancestral to mutation i while $\sum_{p=1}^{m} \hat{f}_{p,i} > \sum_{p=1}^{m} \hat{f}_{p,j}$. Note that as $d_{p,i} \to \infty$ for all p and i, the probability distribution of $a_{p,i}$ is concentrated at $\hat{f}_{p,i} \cdot d_{p,i}$. In other words, as $[d_{p,i}] \to \infty$, we have $\Pr(\hat{F} \mid A, D) = 1$, and $\Pr(F \mid A, D) = 0$ if $F \neq \hat{F}$. Therefore, if T does not adhere to (SC) w.r.t. $\hat{F}$ then there must exists another matrix $F \neq \hat{F}$ such that T adheres to (SC) w.r.t. $F \neq \hat{F}$. However, for this $F \neq \hat{F}$ we would have $\Pr(F \mid A, D) = 0$. This in turn would imply that T is not in $\mathcal{T}^{(\rho)}$ for any $\rho > 0$, proving the lemma. ◀

In practice, entries $D = [d_{p,i}]$ do not go to infinity. Therefore, we consider additional inclusion criteria beyond adding mutation i as a new leaf of T. Specifically, we allow a new mutation i to be a parent of an existing mutation j (cases (i) and (iii)). Since we do not expect the real F to deviate too much from $\hat{F}$, it is unlikely that mutation i will be assigned more than one child of mutation j in tree T, allowing us to avoid enumerating all possible redistributions of the original children of j among i and j. This in turn limits the number of maximum likelihood calculations. The pseudocode of the updated procedure FastBackboneEnumeration is given in Algorithm 1, where AddMutationFast(T, i) corresponds to the faster method of adding mutation i to tree T discussed above.

3.2 Expanding a backbone tree into a full tree

Note that solving Backbone Tree Expansion from Reads given an empty tree is equivalent to finding a maximum likelihood tree given reads A, D, which is NP-hard as discussed. In the following, we propose a greedy algorithm to solve this problem heuristically. We employ similar ideas of growing the tree as in the FastBackboneEnumeration algorithm, considering the remaining mutations $[n] \setminus V(T)$ in descending order according to $\hat{F}$. However, rather than keeping a large set of trees that have a high likelihood, we only retain a single tree with the highest likelihood at each iteration. The pseudo-code of this method called GreedyExpansion is given in Algorithm 2.

Algorithm 1 FastBackoneEnumeration(A,D,ℓ,ρ).

1: $(u_1, \ldots, u_n) \leftarrow$ mutations $[n]$ in descending order of $\sum_{p=1}^{m} \hat{f}_{p,u}$
2: $\mathcal{T}_1 \leftarrow \{T\}$ ▷ T is a tree with single node u_1
3: $S_1 \leftarrow \{u_1\}$
4: **for** $k \leftarrow 1$ to $\ell - 1$ **do**
5: $\quad i \leftarrow u_{k+1}$
6: $\quad S_{k+1} \leftarrow S_k \cup \{i\}$
7: $\quad \mathcal{T}_{k+1} \leftarrow \bigcup_{T \in \mathcal{T}_k}$ AddMutationFast(T,i)
8: $\quad L_{\max} \leftarrow \max_{T \in \mathcal{T}_{k+1}} \mathcal{L}(A[S_{k+1}], D[S_{k+1}] \mid T)$
9: $\quad \mathcal{T}_{k+1} \leftarrow \{T \in \mathcal{T}_{k+1} \mid \mathcal{L}(A[S_{k+1}], D[S_{k+1}] \mid T) \geq L_{\max} + \log(\rho)\}$
10: **end for**
11: **return** $(S_\ell, \mathcal{T}_\ell)$

Algorithm 2 GreedyExpansion(A,D,T).

1: $(u_1, \ldots, u_n) \leftarrow$ mutations $[n]$ in descending order of $\sum_{p=1}^{m} \hat{f}_{p,u}$
2: $S = \{u_1, \ldots, u_\ell\}$
3: **for** $k \leftarrow \ell$ to $n - 1$ **do**
4: $\quad S \leftarrow S \cup \{u_i\}$
5: $\quad \mathcal{T} \leftarrow$ AddMutationFast(T,u_{k+1})
6: $\quad T \leftarrow \arg\max_{T \in \mathcal{T}} \mathcal{L}(A[S], D[S] \mid T)$
7: **end for**
8: **return** T

3.3 Implementation details

Sapling provides a Python implementation of the FastBackboneEnumeration and GreedyExpansion algorithms. As mentioned above, computing $-\mathcal{L}(A, D \mid T)$ is equivalent to solving a convex optimization problem with a convex objective function subject to linear constraints. Therefore, Sapling use the Python package `cvxopt` to optimize $-\mathcal{L}(A, D \mid T)$ efficiently. In line 8 of FastBackoneEnumeration (Algorithm 1), Sapling uses an error tolerance to account for floating point errors and the possibility of underestimating the likelihood. If clusters of mutations are provided rather than individual mutations, Sapling takes the median depth of all mutations in the cluster as the depth of the cluster and the median depth times the average frequency $\hat{F}$ as the variant reads (rounded to the nearest integer) for the cluster and treat the cluster as a single mutation. Sapling is available at `https://github.com/elkebir-group/Sapling.git` under the 3-Clause BSD open source license.

4 Results

In this section, we evaluate Sapling on (i) a set of small simulation instances with known optimal solutions, (ii) a set of larger simulation instances and (iii) real data. Specifically, we compare Sapling's backbone trees (obtained via FastBackboneEnumeration) to the optimal backbone tree sets and to alternative summarization obtained by the MCT algorithm [1]. Moreover, we compare Sapling's expanded trees (obtained via GreedyExpansion) to trees sampled by Pairtree [18, 29] and Orchard [19].

4.1 Simulation setup

To generate a simulation instance with a number n of mutations and number m of samples, we start by randomly generating an unrooted, node-labeled tree T with n nodes/mutations using Prüfer sequences [24]. Next, we root the tree T uniformly at random, followed by drawing m samples of fractions of the n clones corresponding to the nodes in the tree from a Dirichlet distribution, ultimately yielding frequency matrix $F = [f_{p,i}]$. For each frequency $f_{p,i}$, the total number $d_{p,i}$ of reads is drawn from a Poisson distribution with mean $\lambda = 100$, simulating an average sequencing depth of $100\times$. Finally, the variant reads $A = [a_{p,i}]$ are each drawn from a binomial distribution with $d_{p,i}$ trials and success probability $f_{p,i}$.

4.2 Sapling identifies near-optimal backbone trees

We generated a simulation dataset of 20 instances with $n = 8$ mutations and $m = 2$ samples. The small number $n = 8$ of mutations allowed us to exhaustively enumerate the $\mathcal{T}^{(0.9)}$ of complete trees with a likelihood that is at most a fraction of $\rho = 0.9$ away from maximum likelihood. To accomplish this, we generated all $n^{n-1} = 8^7 = 2{,}097{,}152$ trees T and computed their likelihoods $\mathcal{L}(A, D \mid T)$. We ran Sapling's FastBackboneEnumeration algorithm with parameters $\ell \in \{1, \ldots, 8\}$ as well as $\tau \in \{1, 2, 5, 10, 20, 50\}$.

We show the number $|\mathcal{T}^{(0.9)}|$ of trees in Fig. 2a, ranging from 3 to 672 with a median of 65 trees. Next, we enumerated all $2^8 = 256$ subsets $S' \subseteq [n]$ of mutations, and identified the number $|\mathcal{T}^{(0.9)}[S']|$ of backbone trees for each subset S' of mutations. This allowed us to compare the number $|\mathcal{T}^{(0.9)}[S]|$ of backbone trees returned by Sapling for varying values of ℓ to the optimal number $|\mathcal{T}^{(0.9)}[S^*]|$ of backbone trees such that $|S^*| = \ell$ by computing the *approximation ratio* defined as $|\mathcal{T}^{(0.9)}[S]|/|\mathcal{T}^{(0.9)}[S^*]|$. Thus, an approximation ratio of 1 indicates that Sapling identified an optimal (minimum) set of backbone trees. We find the median approximation ratio is 1 with a maximum ratio of 7 with $|\mathcal{T}^{(0.9)}[S]| = 14$ inferred backbone trees by Sapling versus $|\mathcal{T}^{(0.9)}[S^*]| = 2$ optimal backbone trees for $\ell = 5$ mutations (Fig. 2b). In Fig. 2c, we show an instance where Sapling returned optimal solutions for all ℓ, whereas Fig. 2d shows an instance where Sapling did not return optimal solutions for all ℓ. Specifically, for $\ell = 3$ Sapling returned two backbone trees for mutations $S = \{0, 1, 3\}$ shown in Fig. 2e whereas there exists a different set $S^* = \{0, 3, 4\}$ with just a single backbone tree shown in Fig. 2f.

Similarly, we evaluate the approximation ratio when running Sapling with a specified upper bound τ of backbone trees. Specifically, let $|S|$ be the number of mutations returned by Sapling and $|S^*|$ be the maximum number of mutations such that $|\mathcal{T}^{(0.9)}[S]|, |\mathcal{T}^{(0.9)}[S^*]| \leq \tau$. The *approximation ratio* equals $|S|/|S^*|$, where a value of 1 indicates that Sapling returned the optimal (maximum) number of mutations and a value smaller than 1 indicates that Sapling underestimated the number of mutations. Again, we find that the median approximation ratio is 1, with a minimum ratio of 0.25 for which Sapling identified $|S| = 1$ mutations versus a maximum number $|S^*| = 4$ of mutations for $\tau = 1$ (Fig. 2c). In particular, for smaller τ the approximation ratio may be smaller than 1.

Thus, in general, we find that the heuristic employed by Sapling in the majority of cases finds an optimal solution for these small simulation instances. Moreover, the backbone trees returned by Sapling have perfect recall compared to the backbone tree set obtained from the ground-truth complete trees using the same set of mutations (data not shown).

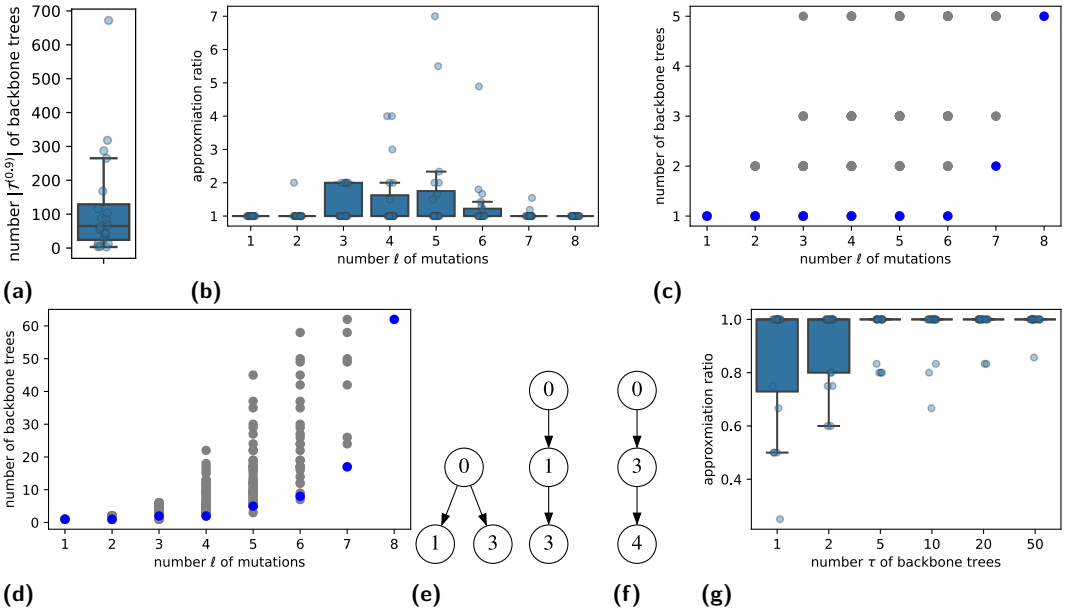

Figure 2 Simulations on $n = 8$ mutations and $m = 2$ samples. (a) Set $\mathcal{T}^{(0.9)}$ of trees that have a likelihood that is at most a factor of $\rho = 0.9$ away from maximum likelihood. (b) Approximation ratio achieved by Sapling for varying ℓ. (c) A simulation instance that Sapling (blue) solves to optimality for all ℓ, with gray entries indicating subsets of mutations not considered by Sapling. (d) A simulation instance that Sapling did not solve to optimality for $\ell \in \{3, 5, 6\}$. (e) The two backbone trees returned by Sapling for the instance shown in (d) at $\ell = 3$. (f) The optimal backbone tree of the instance in (e) determined by exhaustive enumeration at $\ell = 3$. (g) Approximation ratio achieved by Sapling for varying τ.

4.3 Sapling infers high-quality backbone and full trees

In this section, we demonstrate Sapling is also capable of handling larger input instances. To that end, we generated 60 additional simulation instances with $m = 10$ samples and $n \in \{20, 50, 100\}$ mutations (with 20 instances for each value of n). We ran Sapling on a laptop with 16 GB RAM and an Apple M1 Pro CPU. We show the running time of Sapling's backbone tree enumeration mode in Fig. 3a, showing an exponential increase in running time with increasing number n of mutations and increasing values of the parameter $\tau \in \{1, 5, 10, 20, 50\}$.

We additionally ran Sapling's backbone tree expansion algorithm to expand each identified backbone tree into a full tree for all twenty $n = 50$ simulation instances. We find that the running time ranged from a minimum of 47 seconds when provided a $\tau = 50$ backbone tree (containing 47 mutations) vs. 335 seconds when provided a $\tau = 1$ backbone tree (containing 12 mutations) – see Fig. 3b.

To compare Sapling's complete trees, we also ran Pairtree and Orchard and retained their τ highest likelihood unique trees (we ran these algorithms with default parameters using 4 MCMC chains with 2500 samples each and a burn-in of 1250 samples for Pairtree; and a beam width of $k = 10$, a branching factor of $f = 100$ and 8 parallel instances for Orchard). The comparison of the log-likelihood of one $n = 50$ simulation instance is shown in Fig. 3c and the remaining simulation instances are shown in Fig. S1, Fig. S2 and Fig. S3. We find that Sapling's trees achieved higher likelihoods (median: -6627.12 for $\tau = 50$) than those identified by Pairtree (median: -6647.23 for $\tau = 50$) and Orchard (median: -6639.16 for

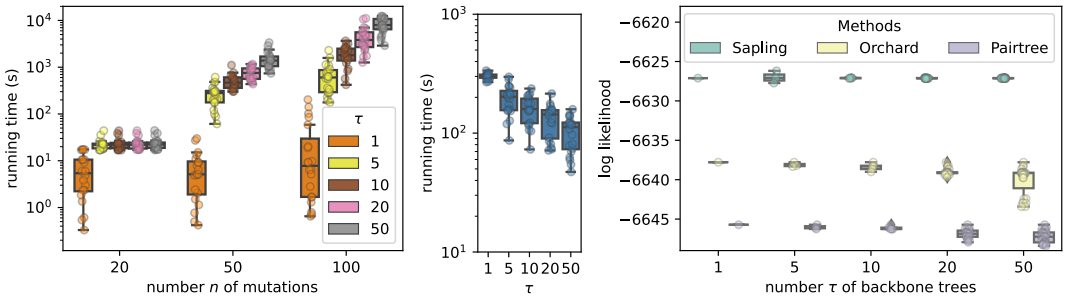

Figure 3 Simulations on $n \in \{20, 50, 100\}$ mutations and $m = 10$ samples. (a) Running time of Sapling's backbone tree enumeration algorithm for varying values of τ. (b) Running time of Sapling's backbone tree expansion algorithm when given an initial backbone tree obtained using the specified τ parameter for simulation instances with $n = 50$ mutations. (c) Likelihood values of complete trees identified by Sapling, Orchard and Pairtree for a single $n = 50$ simulation instance out of 20 simulations.

$\tau = 50$). It is important to note, however, that Pairtree and Orchard (79×4 and 59×8 seconds, respectively) ran much faster than Sapling (2 hours and 30 minutes for backbone enumeration and 31 minutes for backbone expansion for $\tau = 50$).

In summary, Sapling can be used to obtain a diverse set of high-likelihood trees by expanding initial backbone trees.

4.4 Sapling summarizes the solution space of real data

Finally, we ran Sapling on the TRACERx cohort of 100 non-small-cell lung cancer patients [15] using the mutation clusters reported by the authors, whose number n of clusters ranged from 2 to 15 and number m of sequencing samples ranged from 1 to 7. For each patient, we ran Sapling with parameters $\rho \in \{0.4, 0.9\}$ and $\ell \in \{1, \ldots, n\}$. Sapling's running time ranged from less than 1 second to 90 seconds (Apple M1 Pro CPU with 16 GB RAM, data not shown).

Setting $\ell = n$ results in Sapling enumerating the complete solution space $\mathcal{T}^{(\rho)}$ for the specified value of ρ, indicating the allowed deviation from maximum-likelihood. The distribution of $\mathcal{T}^{(\rho)}$ is shown in Fig. 4a, showing that the number of trees increased with decreasing τ as expected. Specifically, there are 26 and 14 patients with at least two trees in $\mathcal{T}^{(\rho)}$ for $\rho = 0.4$ and $\rho = 0.9$, respectively.

On the other hand, when setting $\tau = 1$, Sapling seeks to identify a single backbone tree with a maximum number of mutations. In Fig. 4b, we show the fraction of mutations that are included in each individual backbone tree per patient, finding that a median fraction of 0.75 and 0.81 of mutations is included for $\rho = 0.4$ and $\rho = 0.9$, respectively. We show the $\tau = 1$ backbone tree identified by Sapling with $\rho = 0.4$ for patient CRUK0013 with $n = 9$ mutation clusters. This backbone tree spans 7 out of 9 mutation clusters, and is a proper subtree of the single consensus tree for CRUK0013 reported by the MCT algorithm [1]. For patient CRUK0037 with $n = 10$ mutation clusters, restricting the number of backbone trees to $\tau = 1$ results in only 5 and 6 covered mutation clusters for $\rho = 0.4$ and $\rho = 0.9$, respectively (Fig. 4d). With $\tau = 2$ backbone trees, Sapling covers an additional mutation cluster for both values of ρ. We show the two backbone trees for $\rho = 0.4$ in Fig. 4e, which, again, form proper subtrees of the two consensus trees reported by the MCT algorithm for this patient [1]. The backbone trees of these two patients identified by sweeping $\ell \in \{1, \ldots, n\}$ are shown in Fig. S4 and Fig. S5.

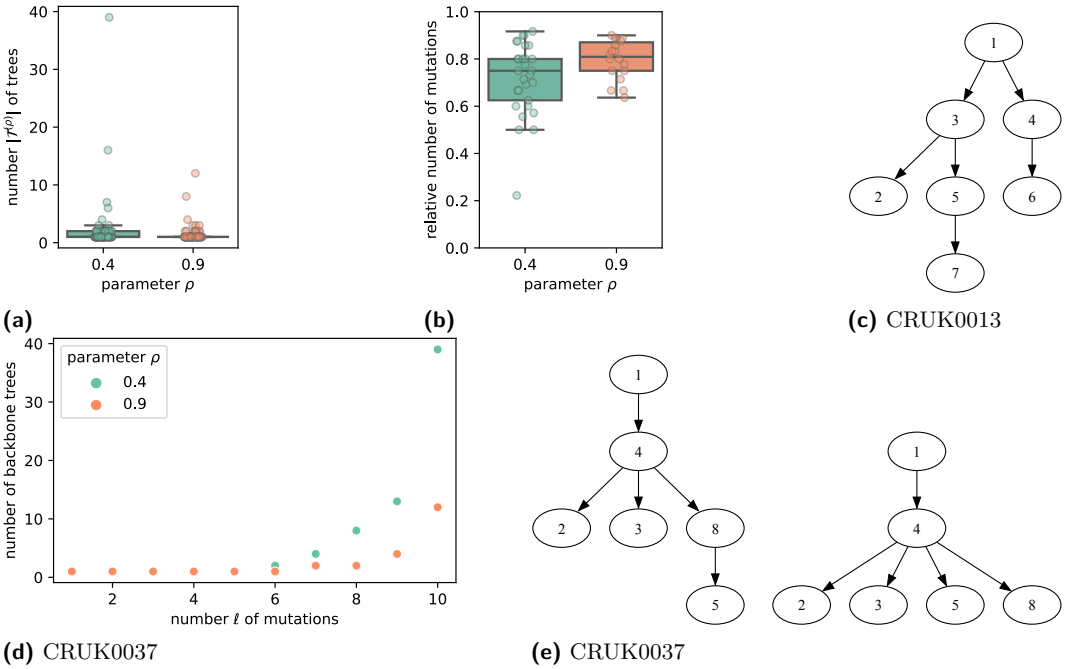

Figure 4 Sapling performance on TRACERx data. (a) The number $|\mathcal{T}^{(\rho)}|$ of complete trees identified by Sapling for $\rho \in \{0.4, 0.9\}$. (b) The fraction of mutation clusters in the backbone tree for $\tau = 1$ (only showing patients where $|\mathcal{T}^{(\rho)}| > 1$). (c) The single backbone tree identified by Sapling with $\tau = 1, \rho = 0.4$ for patient CRUK0013 (see Fig. S4, iteration $\ell = 7$). (d) The number of backbone trees identified by Sapling for varying number ℓ of mutations for patient CRUK0037. (e) The two backbone trees identified by Sapling with $\tau = 2, \rho = 0.4$ for patient CRUK0037 (see Fig. S5, iteration $\ell = 6$).

In summary, on real data, we find that Sapling is able to quickly enumerate the complete solution space of trees and comprehensively summarize it with a small number of backbone trees that span a large fraction of mutations.

5 Discussion

In this work, we introduced the Minimum Cardinality Backbone Trees from Reads and Maximum Mutation Backbone Trees from Reads problems, which seek to identify a set of backbone trees given read count data. These are new problem statements, extending the concept of maximum-agreement subtree (MAST) for leaf-labeled trees [28] to node-labeled trees while allowing for multiple distinct subtrees. In addition, we introduced Backbone Tree Expansion from Reads problem, which seeks to a expand a single backbone tree into a full tree given read count data. We showed that the problems are NP-hard, and introduced a heuristic algorithm, Sapling. Using simulations, we showed that Sapling provides a good approximate solution to both Backbone Trees from Reads problems. We also demonstrated that Sapling returns more plausible trees with higher data likelihoods than the current state-of-the-art methods, Pairtree [18, 29] and Orchard [19]. On real data, we ran Sapling on the TRACERx cohort of 100 lung cancer patients [15], showing that Sapling's backbone trees adequately summarize the solution space of trees.

There are several future directions. First, we could relax the infinite sites assumption and support mutation loss. Second, for $\tau = 1$, where we seek a single backbone tree with maximum number of mutations, we noted a decrease in performance. Therefore, we plan on developing a specialized algorithm for this important case, which has not been studied yet in the literature. Third, we plan to sample from backbone trees rather than expanding these greedily.

References

1 Aguse et al. Summarizing the solution space in tumor phylogeny inference by multiple consensus trees. *Bioinformatics*, 35(14):i408–i416, 2019.

2 Giulio Caravagna, Ylenia Giarratano, Daniele Ramazzotti, Ian Tomlinson, Trevor A Graham, Guido Sanguinetti, and Andrea Sottoriva. Detecting repeated cancer evolution from multi-region tumor sequencing data. *Nature methods*, 15(9):707–714, 2018.

3 Cayley. A theorem on trees. *Quart. J. Math.*, 23:376–378, 1878.

4 Christensen et al. Detecting evolutionary patterns of cancers using consensus trees. *Bioinformatics*, 36(Supplement_2):i684–i691, 2020.

5 Deshwar et al. PhyloWGS: reconstructing subclonal composition and evolution from whole-genome sequencing of tumors. *Genome biology*, 16:1–20, 2015.

6 DiNardo et al. Distance measures for tumor evolutionary trees. *Bioinformatics*, 36(7):2090–2097, 2020.

7 El-Kebir et al. Reconstruction of clonal trees and tumor composition from multi-sample sequencing data. *Bioinformatics*, 31(12):i62–i70, June 2015. doi:10.1093/bioinformatics/btv261.

8 El-Kebir et al. Inferring the mutational history of a tumor using multi-state perfect phylogeny mixtures. *Cell systems*, 3(1):43–53, 2016.

9 Fu and Schwartz. ConTreeDP: A consensus method of tumor trees based on maximum directed partition support problem. In *2021 BIBM*, pages 125–130. IEEE, 2021.

10 Govek et al. A Consensus Approach to Infer Tumor Evolutionary Histories. *BCB*, pages 63–72, 2018.

11 Govek et al. GraphyC: Using consensus to infer tumor evolution. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(1):465–478, 2020.

12 Guang et al. A weighted distance-based approach for deriving consensus tumor evolutionary trees. *Bioinformatics*, 39(Supplement 1):i204–i212, June 2023. doi:10.1093/bioinformatics/btad230.

13 Hodzic et al. Identification of conserved evolutionary trajectories in tumors. *Bioinformatics*, 36(Supplement_1):i427–i435, 2020.

14 Ivanovic and El-Kebir. Modeling and predicting cancer clonal evolution with reinforcement learning. *Genome Research*, 33(7):1078–1088, 2023.

15 Jamal-Hanjani et al. Tracking the evolution of non–small-cell lung cancer. *New England Journal of Medicine*, 376(22):2109–2121, 2017.

16 Karpov et al. A multi-labeled tree dissimilarity measure for comparing "clonal trees" of tumor progression. *Algorithms for Molecular Biology*, 14(1):1–18, 2019.

17 Khakabimamaghani et al. Collaborative intra-tumor heterogeneity detection. *Bioinformatics*, 35:i379–i388, 2019.

18 Kulman et al. Reconstructing cancer phylogenies using pairtree, a clone tree reconstruction algorithm. *STAR protocols*, 3(4):101706, 2022.

19 Ethan Kulman, Rui Kuang, and Quaid Morris. Orchard: building large cancer phylogenies using stochastic combinatorial search. *arXiv preprint arXiv:2311.12917*, 2023.

20 Lohr et al. Widespread genetic heterogeneity in multiple myeloma: Implications for targeted therapy. *Cancer Cell*, 25(1):91–101, January 2014. doi:10.1016/j.ccr.2013.12.015.

21 Malikic et al. Clonality inference in multiple tumor samples using phylogeny. *Bioinformatics*, 31(9):1349–1356, 2015.

22 Nemirovski. Interior point polynomial time methods in convex programming. *Lecture notes*, 42(16):3215–3224, 2004.

23 Nowell. The clonal evolution of tumor cell populations. *Science*, 194(4260):23–8, October 1976.

24 Prufer. Neuer beweis eines satzes uber per mutationen. *Archiv der Mathematik und Physik*, 27:742–744, 1918.

25 Qi et al. Implications of non-uniqueness in phylogenetic deconvolution of bulk DNA samples of tumors. *Algorithms for Molecular Biology*, 14(1):19, December 2019. doi:10.1186/s13015-019-0155-6.

26 Schwartz and Schäffer. The evolution of tumour phylogenetics: principles and practice. *Nature Reviews Genetics*, 18(4):213–229, April 2017. doi:10.1038/nrg.2016.170.

27 Sundermann et al. Reconstructing tumor evolutionary histories and clone trees in polynomial-time with submarine. *PLoS CB*, 17(1):e1008400, 2021.

28 Warnow. *Computational phylogenetics: an introduction to designing methods for phylogeny estimation*. Cambridge University Press, 2017.

29 Wintersinger et al. Reconstructing complex cancer evolutionary histories from multiple bulk DNA samples using pairtree. *Blood Cancer Discovery*, 3(3):208–219, 2022.

A Supplementary Results

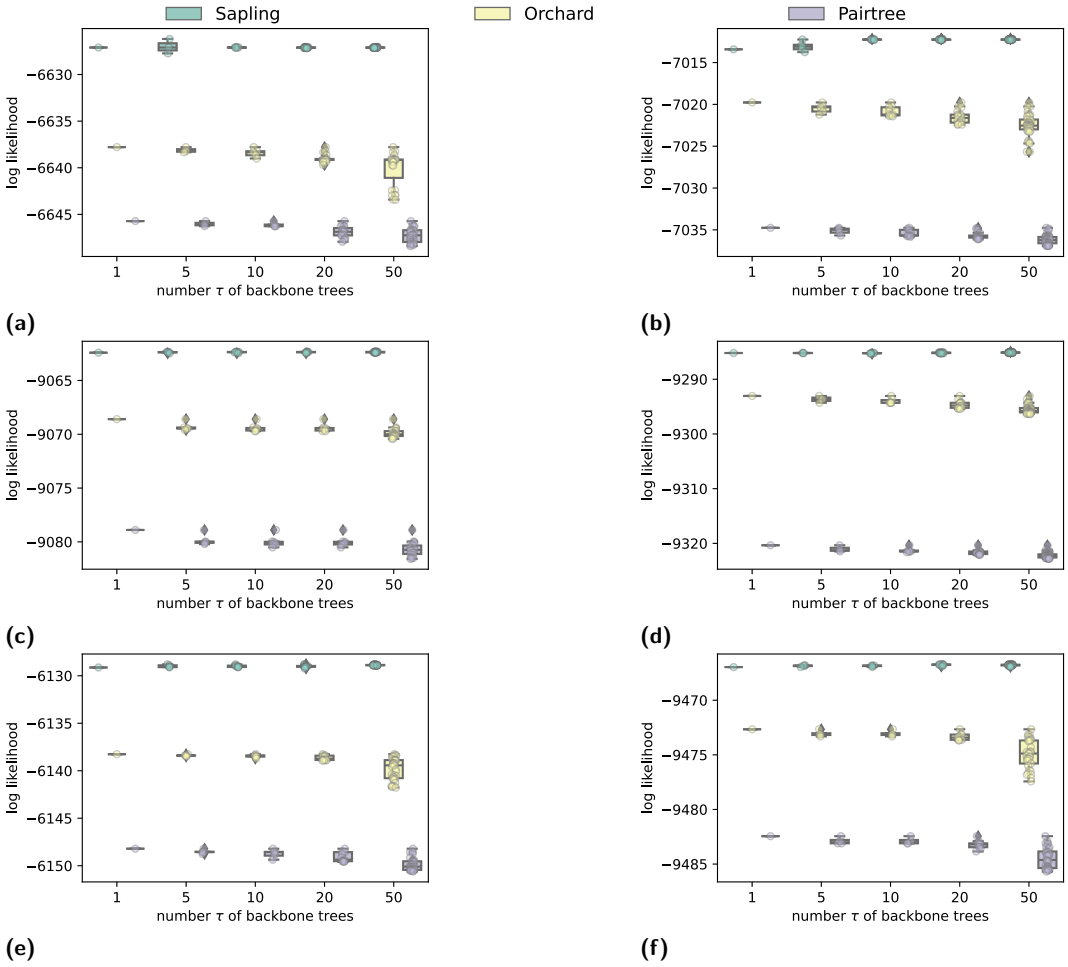

Figure S1 Likelihood values of complete trees identified by Sapling for $n = 50$ simulation instances (with random number generator seeds $1 - 6$).

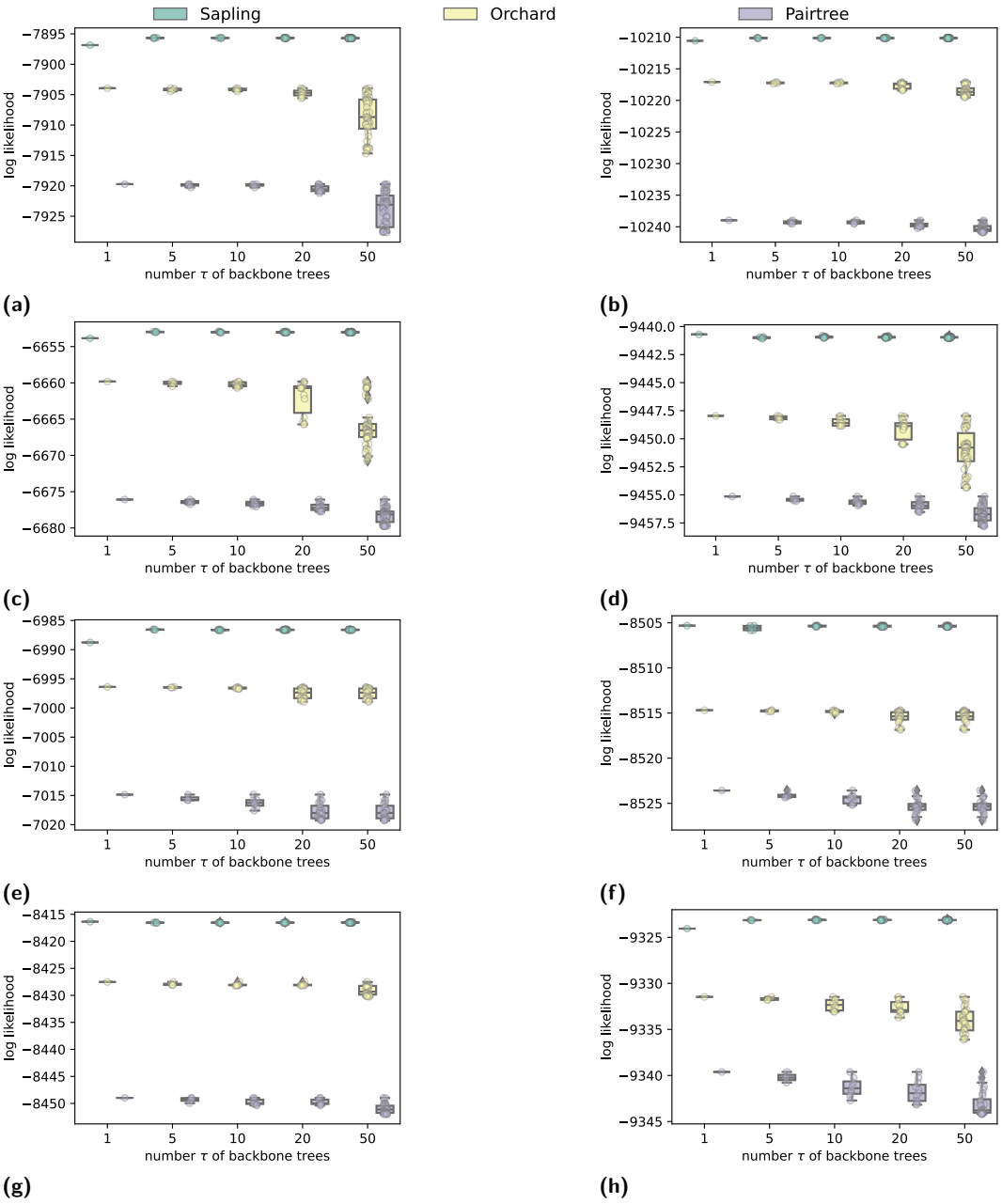

Figure S2 Likelihood values of complete trees identified by Sapling for $n = 50$ simulation instances (with random number generator seeds $7 - 14$).

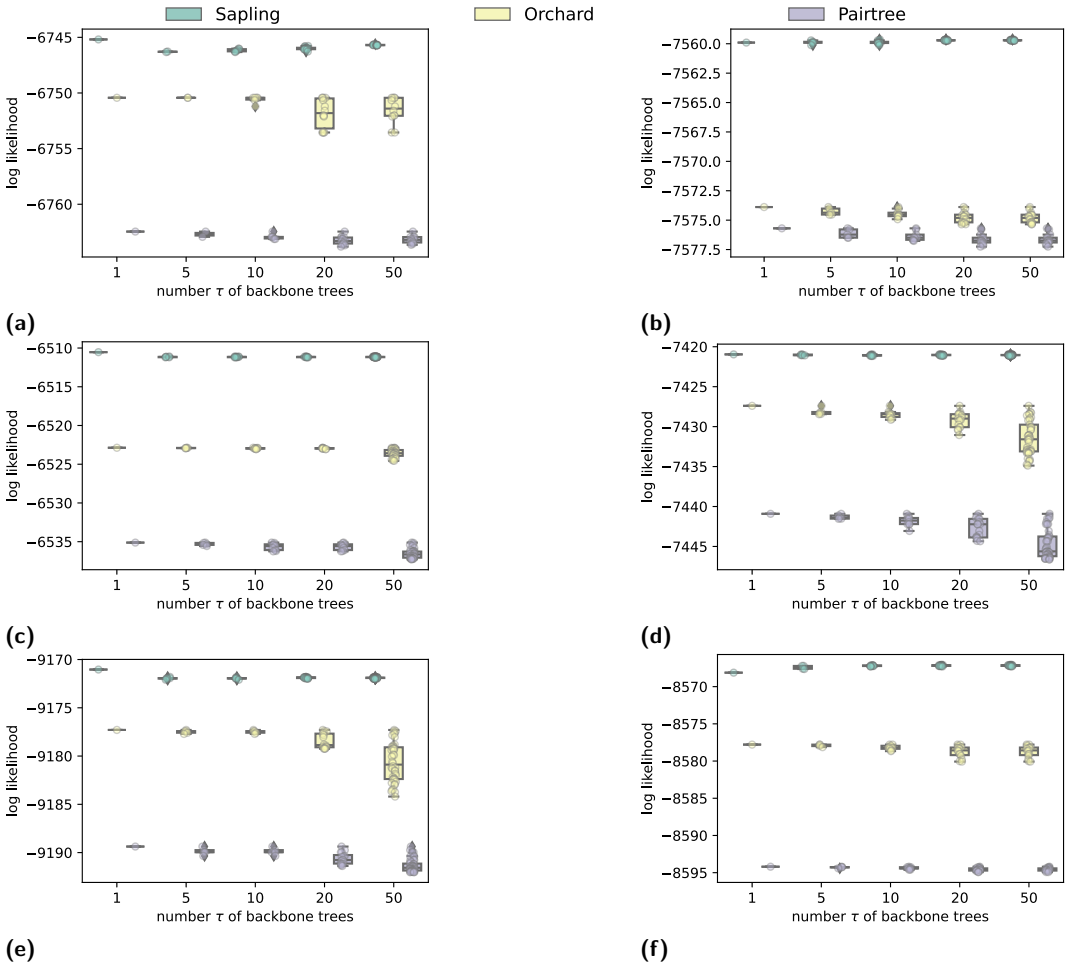

Figure S3 Likelihood values of complete trees identified by Sapling for $n = 50$ simulation instances (with random number generator seeds $15 - 20$).

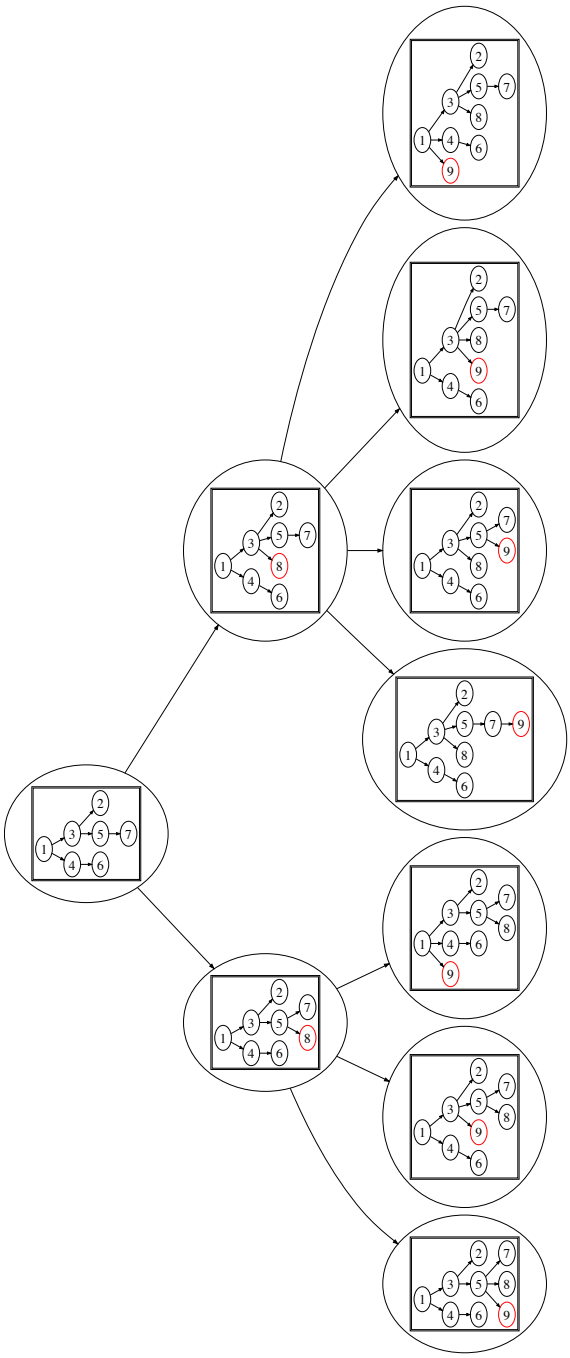

Figure S4 Results of Sapling on TRACERx patient CRUK0013. Sapling was run iteratively starting with $\ell = 1$ and ending with $\ell = n = 9$ mutation clusters and $\rho = 0.4$. Iterations $\ell \in \{1, \ldots, 7\}$ each resulted in a single backbone tree. Note that iteration $\ell = 7$ corresponds to the single backbone tree identified in Fig 4c. Iteration $\ell = 8$ resulted in two backbone trees and the final iteration $\ell = 9$ resulted in seven complete trees. Note that the mutation cluster added in each iteration is indicated in red.

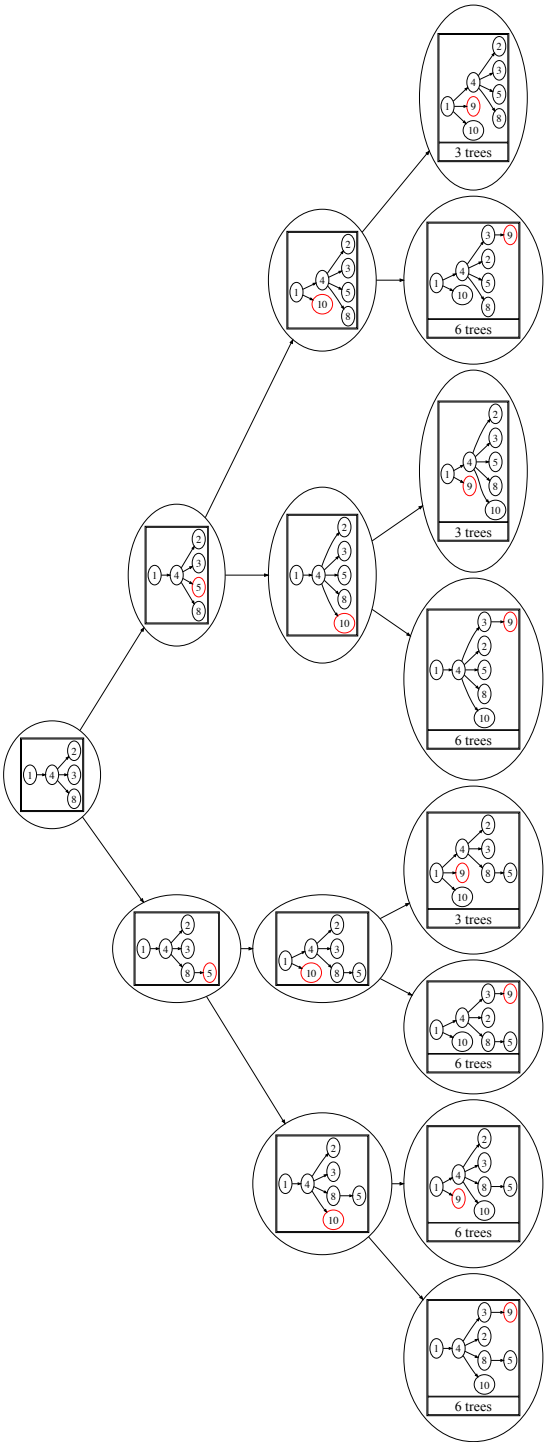

Figure S5 Results of Sapling on TRACERx patient CRUK0037. Sapling was run iteratively starting with $\ell = 1$ and ending with $\ell = n = 10$ mutation clusters and $\rho = 0.4$. Iterations $\ell \in \{1, \ldots, 5\}$ each resulted in a single backbone tree. Iteration $\ell = 6$ resulted in two backbone trees (also shown in Fig. 4e), $\ell = 7$ resulted in 4 backbone trees, and $\ell = 8$ resulted in 8 backbone trees. Due to space constraints, backbone trees identified in iterations $\ell \in \{9, 10\}$ are not shown. Rather, we list the number of full trees summarized by each backbone tree with $\ell = 8$ mutation clusters (box). Note that the mutation cluster added in each iteration is indicated in red.