Proceedings of Machine Learning Research vol 99:1–18, 2019 32nd Annual Conference on Learning Theory

1

# Reconstructing Trees from Traces

**Sami Davies** DAVIESS@UW.EDU
*University of Washington*

**Miklos Z. Racz** MRACZ@PRINCETON.EDU
*Princeton University*

**Cyrus Rashtchian** CRASHTCHIAN@ENG.UCSD.EDU
*University of California, San Diego*

**Editors:** Alina Beygelzimer and Daniel Hsu

## Abstract

We study the problem of learning a node-labeled tree given independent traces from an appropriately defined deletion channel. This problem, tree trace reconstruction, generalizes string trace reconstruction, which corresponds to the tree being a path. For many classes of trees, including complete trees and spiders, we provide algorithms that reconstruct the labels using only a polynomial number of traces. This exhibits a stark contrast to known results on string trace reconstruction, which require exponentially many traces, and where a central open problem is to determine whether a polynomial number of traces suffice. Our techniques combine novel combinatorial and complex analytic methods.

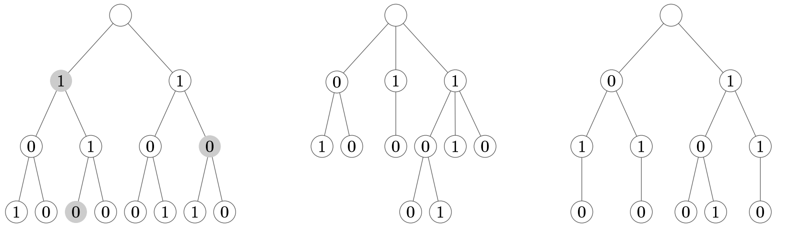
**Keywords:** Trace Reconstruction, Sample Complexity, Deletion Channel

## 1. Introduction

Statistical reconstruction problems aim to recover unknown objects given only noisy samples of the data. In the *string trace reconstruction* problem, there is an unknown binary string, and we observe noisy samples of this string after it has gone through a deletion channel. This deletion channel independently deletes each bit with constant probability $q$ and concatenates the remaining bits. The channel preserves bit order, so we observe a sampled subsequence known as a *trace*. The goal is to learn the original string with high probability using few traces. The string trace reconstruction problem (with insertions and substitutions in addition to deletions) directly appears in the problem of DNA Data Storage [Church et al. (2012); Organick et al. (2018)]. Here it is crucial to minimize the sample complexity, as this directly impacts the cost of retrieving data stored in synthetic DNA.

We introduce a generalization of string trace reconstruction, called tree trace reconstruction. Research on DNA nanotechnology has demonstrated that structures of DNA molecules that are more complex than a line, such as lattices and trees, can be constructed. Also, recent work can distinguish some molecular topologies, such as Y-structures (spiders with three arms), from line DNA using nanopores [Karau and Tabard-Cossa (2018)]. We envision these results may open the door for more complicated tree structures, which could be useful for applications. From a technical perspective, tree trace reconstruction may aid in understanding the interplay of combinatorial and analytic approaches to reconstruction problems and can be a springboard for new ideas.

[*] Full version appears as https://arxiv.org/abs/1902.05101.



($a$) Original Tree ($b$) TED ($c$) Left-Propagation

Figure 1: Deletion models. Gray nodes deleted from original tree (a). Resulting trace in the TED Model (b) and the Left-Propagation Model (c).

Let $X$ be a rooted tree with unknown binary labels on its $n$ non-root nodes. We assume $X$ has a canonical ordering of its nodes, and the children of a node $v$ in $X$ have a left-to-right ordering. The goal of *tree trace reconstruction* is to learn the labels of $X$ with high probability, using the minimum number of traces, knowing only $q$, the deletion model, and the structure of $X$. We consider two deletion models. In both models, each non-root node $v$ in $X$ is deleted independently with constant deletion probability $q$—the root is never deleted—and the resulting tree is called a trace. Also, in both models, deletions are associative, so it suffices to define the behavior of a single deletion.

In the *Tree Edit Distance model*, deletions do not preserve the nodes' degrees (see Figure 1).

- **Tree Edit Distance (TED) model:** When $v$ is deleted, all children of $v$ become children of $v$'s parent. Equivalently, contract the edge between $v$ and its parent, retaining the parent's label. The children of $v$ take $v$'s place as a continuous subsequence in the left-to-right order.

A key motivation for the TED model is that deletions in the TED model correspond exactly to the deletion operation in tree edit distance, which is a well-studied metric for pairs of labeled trees [Bille (2005)]. This metric is relevant for applications, as well; for example, tree edit distance is often used to compare secondary structures of RNA [Zhang and Shasha (1989)].

In contrast, our main motivation for the *Left-Propagation model* is more theoretical: it preserves different structural properties—for instance, a node's number of children does not increase—and poses different challenges than the TED model. To describe this model, we define the *left-only* path starting at $v$ as the path that recursively goes from parent to left-most child, stopping at a leaf.

- **Left-Propagation model:** When $v$ is deleted, recursively replace every node (together with its label) in the left-only path starting at $v$ with its child in the path. This results in the deletion of the last node of the left-only path, with the remaining tree structure unchanged.[1]

Figure 1 depicts example traces in both the TED and the Left-Propagation models, for the same original tree $X$ and the same set of deleted nodes. Note that it may not be clear from a trace which nodes were deleted. Also, observe that when $X$ is a path (with first node as the root) or a star (with center as the root), then both models coincide with the string deletion channel. In many places, we defer to the full version of the paper.

[1] Since the BFS order on $X$ is arbitrary (but fixed), the choice of using the left-only path (as opposed to, say, the right-only one) does not *a priori* bias certain nodes.

### 1.1. Related Work

Introduced by Batu, Kannan, Khanna, and McGregor [Batu et al. (2004)], string trace reconstruction has received a lot of attention, especially recently [De et al. (2017); Hartung et al. (2018); Holden and Lyons (2018); Holenstein et al. (2008); Holden et al. (2018); McGregor et al. (2014); Nazarov and Peres (2017); Viswanathan and Swaminathan (2008)]. Yet there is still an exponential gap between the known upper and lower bounds for the number of traces needed to reconstruct an arbitrary string with high probability and constant deletion probability: it is known that $\exp(O\left(n^{1/3}\right))$ traces are sufficient [De et al. (2017); Nazarov and Peres (2017)] and $\widetilde{\Omega}(n^{5/4})$ traces are necessary [Holden and Lyons (2018)]. Determining whether a polynomial number of traces suffice is a challenging open problem in the area. A well-studied variant is reconstructing a string with random, average-case labels, instead of arbitrary, worst-case labels [Batu et al. (2004); Holden et al. (2018)].

In a few of our algorithms, we reduce various subproblems to the string trace reconstruction problem. Hence, we will use existing results as a black box, and we precisely state the previous results now. Let $T(n, \delta)$ denote the minimum number of traces needed to reconstruct an $n$-bit string with probability at least $1 - \delta$, where the dependence on the deletion probability $q$ is left implicit.

**Theorem 1 (De et al. (2017); Nazarov and Peres (2017))** $T(n, \delta) \leqslant \ln(\frac{1}{\delta}) \cdot e^{Cn^{1/3}}$.

In terms of lower bounds, $T(n, \delta) = \widetilde{\Omega}(n^{1.25})$ for any $\delta$ bounded away from one [Holden and Lyons (2018)]. We discuss related work on other graph reconstruction models in the full version, noting that there are no formal or quantitative connections between these other models and ours.

### 1.2. Our Results

We provide algorithms for two main classes of trees: complete $k$-ary trees and spiders; some results extend beyond these as well. In a *complete $k$-ary* tree, every non-leaf node has exactly $k$ children, and all leaves have the same depth. An $(n, d)$-*spider* consists of $n/d$ paths of $d + 1$ nodes, all starting from the same root. We focus on these two classes because of their varying structure. Spiders behave like a union of disjoint paths, except when some paths have all of their nodes deleted. This allows us to extend methods from string trace reconstruction, with a slightly more complicated analysis. On the other hand, complete $k$-ary trees are so structured that we can use more combinatorial algorithms, which have proven less successful for string trace reconstruction so far. We use *with high probability* to mean with probability at least $1 - O(1/n)$. Also, $[t] := \{1, 2, \ldots, t\}$.

**TED model for complete $k$-ary trees.** Let $X$ be a rooted complete $k$-ary tree along with unknown binary labels on its $n$ non-root nodes. We provide two algorithms to reconstruct $X$, depending on whether the degree $k$ is large or small. We state our theorems in terms of $T(k, \delta)$.

**Theorem 2** *In the TED model, there exist $c, c' > 0$ depending only on $q$ such that if $k \geqslant c \log^2(n)$, then it is possible to reconstruct a complete $k$-ary tree on $n$ nodes with $\exp(c' \cdot \log_k n) \cdot T(k, 1/n^2)$ traces with high probability.*

Theorem 1 implies that $T(k, 1/n^2) = \exp\left(O(k^{1/3})\right)$ if $k \geqslant c \log^2(n)$, so the trace complexity in Theorem 2 is currently $\exp\left(O(\log_k(n) + k^{1/3})\right)$. This is $\mathrm{poly}(n)$ as long as $k = O(\log^3 n)$.

**Theorem 3** *In the TED model, there exists $C > 0$ depending only on $q$ such that $\exp(Ck \log_k n)$ traces suffice to reconstruct a complete $k$-ary tree on $n$ nodes with high probability.*

In particular, when $k$ is a constant, then the trace complexity of Theorem 3 is $\text{poly}(n)$. Theorem 3 makes no restrictions on $k$, but uses more traces than Theorem 2 for $k \geqslant c\log^2 n$.

**Left-Propagation model for complete $k$-ary trees.** We provide two reconstruction algorithms for $k$-ary trees in the Left-Propagation model, leading to the following two theorems. Algorithms, proofs, and details for this model appear in the full version.

**Theorem 4** *In the Left-Propagation model, there exists $c > 0$ depending only on $q$ such that if $k \geqslant c\log n$, then $T(d+k, 1/n^2)$ traces suffice to reconstruct a complete $k$-ary tree of depth $d = O(\log_k n)$ with high probability.*

When $k \geqslant c\log n$, then $d + k = O(k)$, and we reconstruct an $n$-node complete $k$-ary tree with $\exp(O(k^{1/3}))$ traces by using Theorem 1. We also provide an algorithm with no assumptions on $k$.

**Theorem 5** *In the Left-Propagation model, $O(n^\gamma \log n)$ traces suffice to reconstruct an $n$-node complete $k$-ary tree with high probability, where $\gamma = \ln\left(\frac{1}{1-q}\right)\left(\frac{c'k}{\ln n} + \frac{1}{\ln k}\right)$, for a constant $c' > 1$.*

Theorem 5 implies that $\text{poly}(n)$ traces suffice to reconstruct a $k$-ary tree whenever $k = O(\log n)$ and $q$ is a constant. For small enough $q$ and $k$, the algorithm needs only a sublinear number of traces (for example, binary trees with $q < 1/2 - \varepsilon$). As $q$ is a constant, the bound in Theorem 5 can be more simply thought of as $\exp(C' \cdot (d+k))$; and, in Theorem 4 as $\exp(C \cdot (d+k)^{1/3})$.

**Spiders.** The TED and Left-Propagation deletion models are the same for spiders. We provide two reconstruction algorithms, depending on whether the depth $d$ is large or small.

**Theorem 6** *Assume that $d \leqslant \log_{1/q} n$. For $q < 0.7$, there exists $C > 0$ depending only on $q$ such that $\exp(C \cdot d(nq^d)^{1/3})$ traces suffice to reconstruct an $(n,d)$-spider with high probability.*

To understand the statement of this theorem, consider $d = c\log_{1/q} n$ with $c < 1$. A black-box reduction to the string case results in using $\exp(\widetilde{\Omega}(n^{1-c}))$ traces for reconstruction (see the full version for details), whereas Theorem 6 improves this to $\exp(\widetilde{O}(n^{(1-c)/3}))$. Our approach extends previous results based on complex analysis [De et al. (2017); Nazarov and Peres (2017)]. In particular, we analyze a generating function that might be of independent interest, related to Littlewood polynomials.

For large depth $d \geqslant \log_{1/q} n$, full paths of the spider are unlikely to be completely deleted, and in the full version we derive the following result as a corollary of Theorem 1.

**Proposition 7** *For $q < 1$ and all $n$ large enough, an $(n,d)$-spider with $d \geqslant \log_{1/q} n$ can be reconstructed with $2 \cdot T\left(d, \frac{1}{2n^2}\right)$ traces with high probability.*

### 1.3. Overview of TED Algorithms

Previous work on string trace reconstruction mostly utilizes two classes of algorithms: mean-based methods, which use single-bit statistics for each position in the trace, and alignment-based methods, which attempt to reposition subsequences in the traces to their true positions.

Although mean-based algorithms are currently quantitatively better for string reconstruction, they seem difficult to extend to $k$-ary trees under the TED deletion model. Specifically, mean-based methods require a precise understanding of how the bit in position $j'$ of the original tree affects the bit in position $j$ of the trace. For strings, there is a global ordering of the nodes which enables this. Unfortunately, for $k$-ary trees with $k \notin \{1, n\}$ under the TED model, nodes may shift to a variety of locations, making it unclear how to characterize bit-wise statistics. To circumvent this challenge, we provide two new algorithms, depending on whether or not the degree $k$ is large ($k \geqslant c \log^2(n)$). The main idea is to partition the original tree into small subtrees and learn their labels using a number of traces parameterized primarily by $k$ and $\log_k n$, which can be much smaller than $n$.

When $k$ is large enough, we will be able to localize root-to-leaf paths, in the sense that we can identify the location of their non-leaf nodes in the original tree with high probability. By covering the internal nodes of the tree by such paths, we will directly learn the labels for all non-leaf nodes. Then, we observe that the leaves can be naturally partitioned into stars of size $k$, and we can learn their labels by reducing to string trace reconstruction (for strings on $k$ bits).

When $k$ is small, our localization method fails, and we resort to looking at traces which contain even more structure (which requires more traces). We decompose the entire tree into certain subtrees and recover their labels separately. We define a property which is easily detectable among traces and show that when this property holds, we can extract labels for the subtrees that are correct with probability at least 2/3. Then, we take a majority vote to get the correct labels with high probability.

### 1.4. Overview of Spider Techniques

When the paths of a spider are sufficiently long—if they have depth $d \geqslant \log_{1/q} n$—then with probability close to 1, no path is fully deleted in a given trace. This allows us to trivially match paths of the trace spider to paths of the original spider and then use string trace reconstruction algorithms on the individual paths, leading to Proposition 7.

When the paths of a spider are shorter ($d < \log_{1/q} n$), almost all traces have paths fully deleted; here it is unclear which paths were deleted, which forces us to align paths from different traces. We bypass alignment-based methods and use a mean-based algorithm, building off methods introduced in the proof of Theorem 1 by De et al. (2017); Nazarov and Peres (2017). In contrast to strings which are one dimensional, we have the additional difficulty that spiders are two dimensional: one representing which path a node is in, and the other representing where in a path a node is.

## 2. Preliminaries

In what follows, $X$ denotes the (known) underlying tree, along with the (unknown) binary labels on its $n$ non-root nodes. See the full version for standard tree definitions (e.g., depth, ancestor, leaf).

**$k$-ary Tree Algorithm Preliminaries.** Let $X$ be a rooted complete $k$-ary tree with depth $d$. We index the non-root nodes according the BFS order on $X$ (the root is not indexed; the children of the root are $\{0, 1, \ldots, k-1\}$, etc.). We identify nodes of $X$ with their index. For $t \in [d]$, let $\mathcal{J}_t$ be the nodes at depth $t$. Define $\mathcal{I}_1 = \mathcal{J}_1 = \{0, 1, \ldots, k-1\}$, and for $t \geqslant 2$, we set $\mathcal{I}_t = \{i \in \mathcal{J}_t \mid i \bmod k \neq 0\}$. Define $\mathcal{I} = \bigcup_{t=1}^{d-1} \mathcal{I}_t$. We define three unlabeled subtrees of $X$. Let $P_X(i)$ be the path from the root to $i$ in $X$. Define $H_X(i)$ as the union of the left-only path starting at $i$, descending to a leaf $\ell$, and the $k-1$ siblings of $\ell$. Finally, define $G_X(i) = P_X(i) \cup H_X(i)$.

**Canonical subtrees of traces.** We define certain subtrees of a trace, analogous to $P_X(i)$, $H_X(i)$, and $G_X(i)$, and they only depend on the position of $i$ in $X$. We will denote them as $P_Y(i)$, $H_Y(i)$, and $G_Y(i)$. Intuitively, they are subtrees in $Y$ obtained by looking at nodes that should be in the

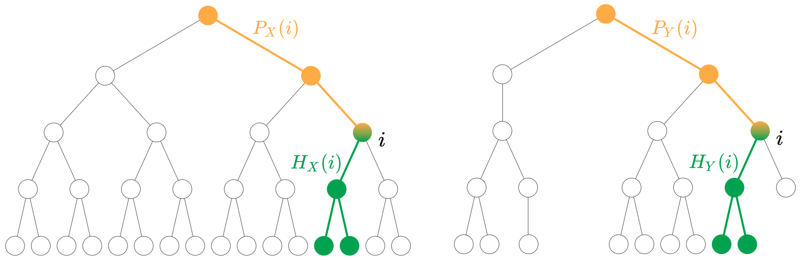


Figure 2: Canonical subtrees for $k$-ary trees, in the original tree (left) and trace (right).

same position as the corresponding ones in $X$. However, the node $i$ does not necessarily belong to these subtrees (e.g., it may have been deleted in $Y$, or another node may be in its place). In what follows, we refer to subtrees as sequences of nodes in the BFS order, since the edge structure will be clear from context (i.e., the subtree is the induced subgraph on the relevant nodes).

We formally define $P_Y(i)$, $H_Y(i)$, and $G_Y(i)$, which are also depicted in Figure 2. Fix $i$, and let $u_0, u_1, \ldots, u_{d-1}$ be the internal nodes in $G_X(i)$, where $u_t$ has depth $t$, and let $u_d, \ldots, u_{d+k-1}$ be the leaf nodes, ordered left-to-right in the BFS order. Define $\pi_i : \{0, 1, \ldots, d-1\} \to \{0, 1, \ldots, k-1\}$ so that $\pi_i(t)$ is the position of $u_{t+1}$ in $X$ among its siblings (the children of its parent $u_t$). Note that $\pi_i$ is independent of the labels of $X$. Let $t_i$ be the depth of $i$ in $X$. We define $P_Y(i)$ as the path $v_0, v_1, \ldots, v_{t_i}$ in $Y$ obtained from the following process. Set $v_0$ to be the root. Then, for $t \in [t_i]$, let $v_t$ be the node at depth $t$ in $Y$ that is in position $\pi_i(t-1)$ among the $k$ children of $v_{t-1}$, where we abort and set $P_Y(i) = \perp$ if $v_{t-1}$ does not have exactly $k$ children. Similarly, let $G_Y(i)$ be the subtree $v_0, v_1, \ldots, v_{d+k-1}$, where $v_t$ is defined as follows. Set $v_0$ to be the root in $Y$. Then, for $t \in [d-1]$, let $v_t$ be the node at depth $t$ in $Y$ that is in position $\pi_i(t-1)$ among the $k$ children of $v_{t-1}$, where we abort and set $G_Y(i) = \perp$ if $v_{t-1}$ does not have exactly $k$ children. Finally, set $v_d, \ldots, v_{d+k-1}$ to be the $k$ children of $v_{d-1}$, and again we set $G_Y(i) = \perp$ if $v_{d-1}$ does not have precisely $k$ children. If $G_Y(i) \neq \perp$, then set $H_Y(i) = v_{t_i}, \ldots, v_{d+k-1}$, and otherwise, set $H_Y(i) = \perp$. Observe that if $G_Y(i) \neq \perp$, then we have $G_Y(i) = P_Y(i) \cup H_Y(i)$.

We remark that $G_Y(i)$, $H_Y(i)$, and $P_Y(i)$ depend only on $\pi_i$ and $Y$, and therefore, they do not use any label information from $X$. We also note that whether these subtrees are set to $\perp$ will be significant, since it implies certain structural properties of traces. If all nodes in $G_X(i)$ survive in a trace $Y$, then we say that $Y$ *contains* $G_X(i)$. We write $G_Y(i) = G_X(i)$ if the nodes and labels in these subtrees are exactly the same (by construction, the edges will also be the same).

## 3. Reconstructing Trees, TED deletion Model

### 3.1. Proof of Theorem 2 concerning large degree trees

Our algorithm utilizes structure that occurs when $k \geqslant c \log^2(n)$. Recall that for a node $i$ in $X$, we think of $i$'s children as being ordered consecutively, left-to-right, based on the BFS ordering of $X$.

**Definition 8** *Let $Y$ be a trace of a tree $X$. We say that $Y$ is $b$-*balanced *if, for every internal node $i$ in $X$, at most $b$ consecutive children of $i$ have been deleted in $Y$.*

**Claim 9** *If $X$ has $n$ nodes, then a trace $Y$ is $b$-balanced with probability at least $1 - nq^b$.*

**Proof** Any set of $b$ consecutive nodes is deleted with probability $q^b$. Since there are at most $n$ starting nodes for a run of $b$ nodes, a union bound proves the claim. ■

We reconstruct $X$ using $b$-balanced traces $Y$. However, since we do not know how to detect whether a trace is balanced or not, we set $b$ large enough so that all traces are balanced with high probability, $b = O(\sqrt{k})$. The balanced structure helps us to determine the position in $X$ of all internal nodes in $Y$ that occur on some surviving root-to-leaf path. For the leaves, we will utilize string trace reconstruction, which applies because the $k$ children of a node at depth $d-1$ are leaves, and the deletion process for a star with $k$ leaves is the same as for the string with $k$ bits.

**Lemma 10** *Let $c, c'$ be large constants depending only on $q$. Assume that $k \geqslant c\log^2(n)$. For a node $j \in \mathcal{J}_{d-1}$, if $Y$ contains the path $P_X(j)$, then there is an algorithm to determine the original position in $X$ of every node of $P_X(j)$ in $Y$ with probability at least $1 - \exp(-c'\sqrt{k})$.*

**Proof** Set $b = 10\sqrt{k}/\log(1/q) = \Omega(\log n)$, so that a trace $Y$ is $b$-balanced with probability at least $1 - \exp(-C'\sqrt{k})$ by Claim 9. In what follows, we assume $Y$ is $b$-balanced. In particular this implies that $Y$ contains some child of every internal node in $X$, because $k > b$ for $k$ large enough (equivalently, $n$ large enough, because $k \geqslant c\log^2(n)$). This property allows us to deduce the depth in $X$ of nodes in $Y$. To see this, let $u$ be any node in $Y$, and let $t$ be the original depth of $u$ in $X$. Then, $Y$ contains a path from $u$ to a leaf in $Y$ with $d - t + 1$ nodes (including $u$). In other words, depth $t$ nodes in $X$ have height $d - t$ in $Y$, when not deleted. As a consequence, $Y$ contains some root-to-leaf path $(u_0, u_1, \ldots, u_d)$ from $X$, and $u_t$ has depth $t$ in both $X$ and $Y$.

Let $\phi : Y \to X$ be the injective function mapping nodes in $Y$ to their positions in $X$. We will determine $\phi(u_t)$ for every $u_t \in P_X(j)$ such that $Y$ contains $P_X(j)$ and $j \in \mathcal{J}_{d-1}$. Without loss of generality, fix $j$ and assume that $P_X(j)$ corresponds to the first $d$ nodes $(u_0, \ldots, u_{d-1})$. Our goal is to verify this fact by determining $\phi(u_t)$ for $u_t \in P_X(j)$.

We know that $u_0$ is the root in both $X$ and $Y$, so consider any depth $t \in [d-1]$ and suppose that we have already determined $\phi(u_{t-1})$. Among the children of $u_{t-1}$ in $Y$, there is a subset that were originally children of $\phi(u_{t-1})$ in $X$. Denote these surviving children as $w_1, \ldots, w_{k'}$, for $1 \leqslant k' \leqslant k$, where we order the $w_i$ from left-to-right in the BFS ordering. We can identify $w_1, \ldots, w_{k'}$ in $Y$, because they will have height $d-t$ in $Y$, while their siblings in $Y$ will have height at most $d-t-1$, which follows from our earlier discussion about consequences of being $b$-balanced. For some $i'$, we have $w_{i'} = u_t$, but we will more generally determine $\phi(w_i)$ for all $i \in [k']$. We will do this by determining the original position of $w_i$ in $X$ among the children of $\phi(u_{t-1})$.

Let $a_i$ be the (currently unknown) number of deleted children of $u_{t-1}$ between $w_i$ and $w_{i+1}$, where we set $a_0$ (resp. $a_{k'}$) to be the number of deleted nodes before $w_1$ (resp. after $w_{k'}$) in $Y$. Observe that $w_i$ has position $i + \sum_{0 \leqslant j < i} a_j$ in $X$ among the children of $\phi(u_{t-1})$. Therefore, our goal will be to determine $a_0, \ldots, a_{k'}$ with high probability.

Let $R_i$ be the total number of surviving descendants in $Y$ of these $a_i$ deleted children. Let $m_t = \sum_{\ell=1}^{d-t} k^\ell$ be the number of edges in a complete $k$-ary tree of depth $d - t$, and observe that each of the $a_i$ deleted children has $m_t$ descendants in $X$, each which survive with probability $(1-q)$ independently. In other words, $R_i$ is a Binomial random variable with $a_i \cdot m_t$ trials and probability $(1 - q)$ of success, and $\mathbb{E}[R_i] = a_i \cdot (1 - q)m_t$.

Consider the event that, for every $i = 0, 1, \ldots, k'$, we have

$$|R_i - a_i \cdot (1 - q) \cdot m_t| \leqslant \frac{(1 - q)m_t}{3}. \tag{1}$$

We claim that Eq. (1) holds with probability at least $1-\exp(-C\sqrt{k})$ by a standard Chernoff bound, since the $R_i$ are Binomial random variables, where we use that $a_i \leqslant b = O(\sqrt{k})$ to bound the deviation of $R_i$. We now argue that if Eq. (1) holds, then we can determine the position of each $w_i$ among the children of $\phi(u_{t-1})$, and so, we can determine $\phi(w_1), \ldots, \phi(w_{k'})$. To achieve this, set $\widehat{a}_i$ to be the unique integer satisfying $\widehat{a}_i - 1/2 \leqslant \frac{R_i}{(1-q)m_t} < \widehat{a}_i + 1/2$. By Eq. (1), we have that $\widehat{a}_i = a_i$ for $i = 0, 1, \ldots, k'$. We deduce that the node $w_i$ has position $i + \sum_{0 \leqslant j < i} \widehat{a}_j$ in $X$ among the children of $\phi(u_{t-1})$, for $i = 0, 1, \ldots, k'$. Therefore, knowing $\phi(u_{t-1})$ and assuming Eq. (1) allows us to determine $\phi(w_i)$ as well. We now put everything together. Any trace $Y$ is $b$-balanced with probability $1-\exp(-O(\sqrt{k}))$. When $Y$ is $b$-balanced and contains $P_X(j)$, we determine the positions of the nodes in this path with probability $1-\exp(-O(\sqrt{k}))$. Although $Y$ may contain $P_X(j)$ for many values of $j \in \mathcal{J}_{d-1}$, there are at most $n$ such paths. Since $k \geqslant c\log^2 n$, we can take a union bound, and we succeed in determining the positions of every $P_X(j)$ in $Y$ with probability at least $1-\exp(-c'\sqrt{k})$ for some constant $c' > 0$ depending only on $q$. ■

**Lemma 11** *Fix $j \in \mathcal{J}_{d-1}$. Using $T(k, 1/n^2)$ traces that each contain $P_X(j)$, we can reconstruct the labels for $P_X(j)$ and all children of $j$ with probability at least $1-2/n^2$.*

**Proof** Consider a trace $Y$ containing $P_X(j)$. Using Lemma 10, we can locate every node of $P_X(j)$ in $Y$ with probability at least $1-\exp(-c'\sqrt{k})$. Finding the labels for $P_X(j)$ is trivial, since the path from the root to $j$ in $Y$ will correspond to the nodes of $P_X(j)$, in order, and these will have the correct labels. For the leaves, we will utilize the string trace reconstruction algorithm (Theorem 1). Indeed, since we have assumed that each trace $Y$ contains $P_X(j)$, we know that the children of $j$ in $Y$ are a subset of the $k$ children of $j$ in $X$. Each one of these leaves is deleted with probability $q$ independently, and they are presented in the same order as a string of length $k$ through the deletion channel. Therefore, Theorem 1 applies, and the $T(k, 1/n^2)$ traces will suffice to reconstruct the labels for the $k$ children of $j$ in $X$ with probability at least $1-1/n^2-\exp(-c'\sqrt{k})$. In conclusion, with probability $1-2/n^2$, we can reconstruct the labels for $P_X(j)$ and children of $j$. ■

**Proof** [Proof of Theorem 2] The path $P_X(j)$ for $j \in \mathcal{J}_{d-1}$ consists of $d$ nodes, so it survives in a trace with probability $(1-q)^d$. Sample $C\,(1-q)^{-d}\,T\left(k, 1/n^2\right)$ traces, where $C$ is a large enough constant to guarantee that with probability at least $1-O(1/n^2)$, we will see at least $T\left(k, 1/n^2\right)$ traces that contain $P_X(j)$. Using Lemma 11, we can reconstruct $P_X(j)$ and all children of $j$ using some $T(k, 1/n^2)$ traces that contain $P_X(j)$ with probability $1-O(1/n^2)$. Applying a union bound over $\mathcal{J}_{d-1}$ with $|\mathcal{J}_{d-1}| \leqslant n$, we learn the labels for all nodes in $X$ with probability $1-O(1/n)$. ■

### 3.2. Proof of Theorem 3 concerning arbitrary degree trees

We will recover the labels for $G_X(i)$ for each $i \in \mathcal{I}$, which is sufficient because these subtrees cover all of the non-root nodes in $X$. The challenge is that $G_X(i)$ may shift to an incorrect position, even when $G_Y(i) \neq \perp$. This happens, for example, when the parent of $i$ has children deleted in such a way that $i$ moves to the left or right, but $i$ still has $k-1$ siblings (some of which are new).

Let $u$ be a node in $G_X(i)$ with child $u'$ that is not a leaf (so $u$ and $u'$ both originally have $k$ children). If $u$ and all of its $k$ children survive in a trace, then we will be in good shape. However, consider the situation when $u$ survives and $u'$ is deleted. In the TED model, we expect $(1-q)k$

children of $u'$ to move up to become children of $u$. The bad case is when $u$ has exactly $k$ children in a trace after some of its original children are deleted. This only happens when subtrees rooted at children of $u$ are completely deleted. If such a subtree is large ($u$ is higher up in the tree), then this is extremely unlikely. We use the following property to force the relevant subtrees to survive.

**Definition 12** *A trace $Y$ is $s$-stable for $i \in \mathcal{I}$ if $G_Y(i) \neq \perp$, and for every internal node $v$ in $G_Y(i)$ with height $h \leqslant s$ in $Y$, each of the $k$ children of $v$ has height exactly $h-1$ in $Y$.*

An obvious way for $Y$ to be $s$-stable is for it to contain $G_X(i)$ and enough relevant descendants of nodes in $G_X(i)$. Let $G_X^+(i)$ be the union of $G_X(i)$ and the $k$ children of every internal node in $G_X(i)$. Then $Y$ will be $s$-stable if it contains $G_X^+(i)$ and at least one path to a leaf (in $X$) from every node in $G_X^+(i)$ with height at most $s$. In Lemma 13, we even argue that this happens with high enough probability to achieve the bound in the theorem. Unfortunately, we cannot directly check whether $Y$ contains the exact nodes in $G_X^+(i)$. We can check if $Y$ is $s$-stable for $i$ by examining the nodes of $G_Y(i)$ and their descendants in $Y$. But if $Y$ is $s$-stable, then it is still not necessarily the case that $G_Y(i) = G_X(i)$, since the nodes in $G_X(i)$ may have shifted in $Y$ or been deleted. To get around this complication, we rely on the $s$-stable property of a trace. We argue in Lemma 14 that if $s$ is large enough and a trace $Y$ is $s$-stable for $i$, then with probability at least 2/3, we have $G_Y(i) = G_X(i)$. Taking a majority vote of $G_Y(i)$ over $O(\log n)$ traces, we recover $G_X(i)$ with high probability. We fix $s = \left\lceil \log_k \log_{1/q}(3dk) \right\rceil$. The proofs of the next two lemmas are in Appendix A.

**Lemma 13** *For $i \in \mathcal{I}$, a trace is $s$-stable for $i$ with probability at least $(1-q)^{dk+s^2k}$.*

**Lemma 14** *If $Y$ is an $s$-stable trace for $i$, then $G_Y(i) = G_X(i)$ with probability at least $2/3$.*

**Proof** [Proof of Theorem 3] (sketch) Let $\mathcal{A}$ be a set of $T = C\log(n)/(1-q)^{dk+s^2k}$ traces with $C$ a large enough constant. By Lemma 13, each trace in $\mathcal{A}$ is $s$-stable for $i$ with probability $(1-q)^{dk+s^2k}$. Therefore, by setting $C$ large enough and taking a union bound over $i \in \mathcal{I}$, we can ensure that with probability at least $1 - 1/n^2$, there is $\mathcal{A}_i \subseteq \mathcal{A}$ that of $s$-stable traces for $i$ with $|\mathcal{A}_i| \geqslant C' \log n$, for every $i \in \mathcal{I}$. By Lemma 14, each trace $Y \in \mathcal{A}_i$ has the property that $G_Y(i) = G_X(i)$ with probability at least 2/3. Let $f_i(Y) \in \{0,1\}^{d+k-1}$ be the labels of $G_Y(i)$ in $Y$. In expectation over $Y \in \mathcal{A}_i$, we have that at least a 2/3 fraction of $Y$ satisfy $f_i(Y) = f_i(X)$. Since $|\mathcal{A}_i| \geqslant C' \log n$ for a large enough constant $C'$, we have by a Chernoff bound that the majority value of $f_i(Y)$ over $Y \in \mathcal{A}_i$ is equal to $f_i(X)$, with probability at least $1 - 1/n^2$. For each $i \in \mathcal{I}$, our reconstruction algorithm will use this majority vote to deduce the labels for $G_X(i)$. Taking a union bound over $i \in \mathcal{I}$, where $|\mathcal{I}| \leqslant n$, we correctly label all nodes with probability at least $1 - 2/n$. To show that $T = \exp(O(dk))$, where $d = O(\log_k n)$, we simply plug in $s = \left\lceil \log_k \log_{1/q}(3dk) \right\rceil$. ■

## 4. Reconstructing Spiders

In the regime where spiders have short paths ($d \leqslant \log_{1/q} n$), we use mean-based algorithms that generalize the methods of De et al. (2017); Nazarov and Peres (2017).

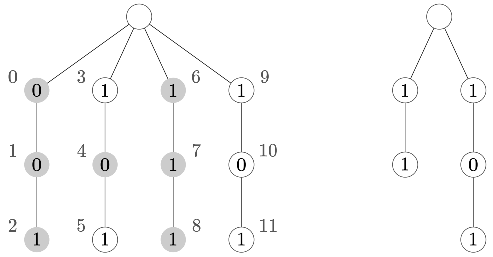


Figure 3: DFS indexing and example trace (in both deletion models) for a $(12,3)$-spider.

**Spider Preliminaries.** When a labeled $(n,d)$-spider, $X$, goes through the deletion channel, we assume that its trace, $Y$, is an $(n,d)$-spider by inserting nodes labeled 0s after the remaining paths and nodes. After this, all traces have $n/d$ paths of length $d$. We define a left-to-right ordered DFS index for $(n,d)$-spiders, illustrated in Figure 3. The labels increase along the length of the paths from the root and increase left to right among the paths. Specifically, if node $v$ is in the $i$th path from the left and has depth $j$, then its label is $(i-1)d+j-1$. These labels will be used to define appropriate generating functions. Since the root is not deleted, it is not considered as part of the generating function. When $d$ is constant, the reconstruction problem on $(n,d)$-spiders can be reduced to string trace reconstruction (see the full version). In what follows, we assume that $d \geqslant 20$.

### 4.1. Proof of Theorem 6 concerning $(n,d)$-spiders with small $d$

We compute the expected generating function for an $(n,d)$-spider that has gone through a deletion channel with parameter $q$. We denote this expected generating function by $A(w)$, where $w \in \mathbb{C}$.

**Lemma 15** *Let $a = \{a_i\}_{i=0}^{n-1}$ be the labels of an $(n,d)$-spider with labels $a_i \in \mathbb{R}$ and let $b = \{b_j\}_{j=0}^{n-1}$ be the labels of its trace from the deletion channel with deletion probability $q$. Then*

$$A(w) := \mathbb{E}\left(\sum_{j=0}^{n-1} b_j w^j\right) = (1-q)\sum_{\ell=0}^{n-1} a_\ell (q+(1-q)w)^{\ell \pmod d} \left(q^d + (1-q^d)w^d\right)^{\lfloor \frac{\ell}{d} \rfloor},$$

*where the expectation is over the random labels $b$.*

While $A(w)$ is written as only a function of $w$, it implicitly depends on the labels $a$ of the original spider. We use this generating function to distinguish between two candidate $(n,d)$-spiders $X^1$ and $X^2$, which have labels $a^1 = \{a_j^1\}_{j=0}^{n-1}$ and $a^2 = \{a_j^2\}_{j=0}^{n-1}$ which are different (that is, there exists $j \in \{0,1,\dots,n-1\}$ such that $a_j^1 \neq a_j^2$). Let $Y^1$ and $Y^2$ denote random traces with labels $b^1 = \{b_j^1\}_{j=0}^{n-1}$ and $b^2 = \{b_j^2\}_{j=0}^{n-1}$ that arise from passing $X^1$ and $X^2$ through the deletion channel with deletion probability $q$. Define $a := a^1 - a^2$ and let $A(w)$ be the expected generating function with input $a$. From Lemma 15 we have that

$$\sum_{j=0}^{n-1} \left(\mathbb{E}\left[b_j^1\right] - \mathbb{E}\left[b_j^2\right]\right) w^j = A(w). \tag{2}$$

Let $\ell^* := \arg\min_{\ell \geqslant 0} \{a_\ell \neq 0\}$ (note that $\ell^* \leqslant n-1$). We can write $A(w) = (q^d + (1-q^d)w^d)^{\lfloor \frac{\ell^*}{d} \rfloor} \cdot \widetilde{A}(w)$, where we call $\widetilde{A}(w)$ the factored generating function. Taking absolute values in Eq. (2),

$$\sum_{j=0}^{n-1} \left|\mathbb{E}\left[b_j^1\right] - \mathbb{E}\left[b_j^2\right]\right| |w|^j \geqslant |A(w)| = (1-q)\left|(1-q^d)w^d + q^d\right|^{\lfloor \frac{\ell^*}{d} \rfloor} \left|\widetilde{A}(w)\right|. \tag{3}$$

Ultimately, we aim to bound from below $\max_j \left|\mathbb{E}\left[b_j^1\right] - \mathbb{E}\left[b_j^2\right]\right|$ by choosing $w \in \mathbb{C}$ appropriately. We consider points on the arc $\gamma_L := \{e^{i\theta} : -\pi/L \leqslant \theta \leqslant \pi/L\}$, where $L \geqslant 20$. The following lemmas are needed to bound the generating function (see the full version for the proof of Lemma 17 and Appendix B for the proofs of the other lemmas in this section).

**Lemma 16** *For $w \in \gamma_L$ we have that* $\left|(1-q^d)w^d + q^d\right| \geqslant \exp\left(-2\pi^2 \cdot q^d(1-q^d)d^2/L^2\right)$.

**Lemma 17** *Let $0 < q < 0.7$ be a constant. There exists $\zeta \in \gamma_L$, as well as a constant $C > 0$ depending only on q, such that $|\widetilde{A}(\zeta)| \geqslant \exp\left(-C \cdot dL\right)$.*

**Proof** [Proof outline of Lemma 17] Let $\Omega \subset \mathbb{C}$ be a bounded, open region, and let $\partial\Omega$ denote its boundary. The *harmonic measure* of a subset $\gamma \subset \partial\Omega$ with respect to a point $w_0 \in \Omega$, will be denoted by $\mu_\Omega^{w_0}(\gamma)$. Let $f(w)$ denote an analytic function; we will choose $f = \widetilde{A}$. We know that $\log|f|$ satisfies the *sub-mean value property*: for all $w_0 \in \Omega$ we have that

$$\log|f(w_0)| \leqslant \int\limits_{\partial\Omega} \log|f(w)| d\mu_\Omega^{w_0}(w). \tag{4}$$

As in Eq. (4), we will define a region of integration where the value of $\log\left|\widetilde{A}(w)\right|$ is controlled along the boundary, and the boundary will contain $\gamma_L = \{e^{i\theta} : -\pi/L \leqslant \theta \leqslant \pi/L\}$. In fact, the methods of Hartung et al. (2018) show a lower bound for $\sup_{\gamma_L} |f(w)|$ for an analytic function $f(w)$ satisfying the growth condition in Lemma 18, by using Eq. (4) and a particular choice of $w_0$.

**Lemma 18** *For all $w \in \mathbb{D}$ and all deletion probabilities $q \in (0,1)$, we have* $\left|\widetilde{A}(w)\right| \leqslant \frac{1}{(1-q)(1-|w|)}$.

The crucial insight is that $\widetilde{A}$ also satisfies the growth condition specified in Lemma 18, allowing us to borrow methods from Hartung et al. (2018) to upper bound the right hand side of Eq. (4). However, we have to work more to find an appropriate point $w_0 \in \mathbb{D}$ in order to find a lower bound for the left hand side of Eq. (4), so that we can also show a lower bound for $\sup_{\gamma_L} |\widetilde{A}(w)|$. ■

**Proof** [Proof of Theorem 6] Let $\zeta \in \gamma_L$ be the point guaranteed by Lemma 17. Substituting $\zeta$ into Eq. (3) (and dropping the factor of $1 - q^d$), we use Lemma 17 and Lemma 16 to see that

$$\sum_{j=0}^{n-1} \left|\mathbb{E}\left[b_j^1\right] - \mathbb{E}\left[b_j^2\right]\right| \geqslant |A(\zeta)| \geqslant (1-q)\exp\left(-2\pi^2 \cdot q^d nd/L^2\right)\exp\left(-C \cdot dL\right),$$

for a constant $C > 0$ depending only on $q$. Setting $L = \max\{\left(4\pi^2 nq^d/C\right)^{1/3}, 20\}$ and plugging into the display above, we find that there exists an index $j$ such that

$$\left|\mathbb{E}\left[b_j^1\right] - \mathbb{E}\left[b_j^2\right]\right| \geqslant \frac{1}{n}\exp\left(-C' \cdot d(nq^d)^{1/3}\right) \tag{5}$$

for some constant $C' > 0$ depending only on $q$. Therefore, we have shown that there is some index $j = j\left(X^1, X^2\right)$ where we expect the traces corresponding to $X^1$ and $X^2$ to differ significantly.

Suppose spider $X^1$ goes through the deletion channel and we observe $T$ samples, $S^1, \ldots, S^T$ where sample $S^t$ has labels $\{u_j^t\}_{j=0}^{n-1}$. Let $\eta$ denote the right hand side of Eq. (5). We say that a spider $X^2$ is a *better match* than $X^1$ for traces $\{S^t\}_{t\in[T]}$ if at the index $j = j(X^1, X^2)$, $X^2$ looks closer to the traces than $X^1$; that is, if

$$\left|\frac{1}{T}\sum_{t=1}^{T} u_j^t - \mathbb{E}\left[b_j^2\right]\right| \leqslant \left|\frac{1}{T}\sum_{t=1}^{T} u_j^t - \mathbb{E}\left[b_j^1\right]\right|.$$

As before, the expectation is over the random labels $b^1$ and $b^2$. A Chernoff bound implies that if the traces $\{S^t\}_{t\in[T]}$ came from spider $X^1$, then the probability that $X^2$ is a better match than $X^1$ is at most $\exp(-T\eta^2/2)$. Repeating this for all pairs of binary labeled $(n, d)$-spiders, the algorithm outputs $X^*$, the $(n, d)$-spider which is a better match than all others (the best match), if such a spider exists. Otherwise, the algorithm outputs a random binary labeled $(n, d)$-spider.

We bound from above the probability that the algorithm does not find that $X^1$ is the best match by a combination of a union bound and a Chernoff bound (as discussed above). The probabilities below are taken over the random traces $\{S^t\}_{t\in[T]}$:

$$\begin{aligned}\mathbf{Pr}[X^* \neq X^1] &\leqslant \sum_{X^2 : X^2 \neq X^1} \mathbf{Pr}[X^2 \text{ is a better match than } X^1] \leqslant 2^n \cdot \exp\left(-T\eta^2/2\right) \\ &= 2^n \exp\left(-\frac{T}{2n^2}\exp\left(-C\cdot d(nq^d)^{1/3}\right)\right)\end{aligned}$$

for $C > 0$ depending only on $q$. This latter expression is at most $1/n$ if $T \geqslant \exp\left(cd\left(nq^d\right)^{1/3}\right)$ for a large enough constant $c$ depending only on $q$. ■

## 5. Conclusions and Future Directions

We introduced the problem of tree trace reconstruction, and we demonstrated, for multiple classes of trees, that we can utilize the structure of trees to develop more efficient algorithms than the current state-of-the-art for string trace reconstruction.

Our paper leaves open many problems and initiates several directions for future work. For one, can our existing sample complexity bounds be improved? Of particular interest are (1) the TED model for complete $k$-ary trees with $\omega(1) \leqslant k \leqslant c\log^2 n$ and (2) spiders with depth $d = c\log_{1/q} n$, $c < 1$; can we reconstruct with $\mathrm{poly}(n)$ traces in these cases? More generally, what is the sample complexity for other classes of trees? What properties of the tree structure are most relevant for reconstructing with fewest traces? Finally, we have focused on deletion channels, but insertions and substitutions are well-defined and relevant for tree edit distance applications. It would be worthwhile to understand the sample complexity for these edits as well.

### 5.1. Acknowledgments

We thank Nina Holden for helpful discussions relating to Lemma 17 and Bichlien Nguyen and Karin Strauss for pointing us to connections on branched DNA and recent work in this area. We also thank Alyshia Olsen for help designing the figures.

## References

Tugkan Batu, Sampath Kannan, Sanjeev Khanna, and Andrew McGregor. Reconstructing strings from random traces. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 910–918, 2004. URL http://dl.acm.org/citation.cfm?id=982792.982929.

Philip Bille. A survey on tree edit distance and related problems. *Theor. Comput. Sci.*, 337(1-3): 217–239, 2005. doi: 10.1016/j.tcs.2004.12.030. URL https://doi.org/10.1016/j.tcs.2004.12.030.

George M. Church, Yuan Gao, and Sriram Kosuri. Next-Generation Digital Information Storage in DNA. *Science*, 337(6102):1628, 2012.

Anindya De, Ryan O'Donnell, and Rocco A. Servedio. Optimal mean-based algorithms for trace reconstruction. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1047–1056, 2017. doi: 10.1145/3055399.3055450. URL http://doi.acm.org/10.1145/3055399.3055450.

Lisa Hartung, Nina Holden, and Yuval Peres. Trace reconstruction with varying deletion probabilities. In *Proceedings of the Fifteenth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 54–61, 2018. doi: 10.1137/1.9781611975062.6. URL https://doi.org/10.1137/1.9781611975062.6.

Nina Holden and Russell Lyons. Lower bounds for trace reconstruction. Preprint available at https://arxiv.org/abs/1808.02336, 2018.

Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. In *Proceedings of the 31st Conference On Learning Theory (COLT)*, pages 1799–1840, 2018. URL http://proceedings.mlr.press/v75/holden18a.html.

Thomas Holenstein, Michael Mitzenmacher, Rina Panigrahy, and Udi Wieder. Trace reconstruction with constant deletion probability and related results. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 389–398, 2008. URL http://dl.acm.org/citation.cfm?id=1347082.1347125.

Phillip Karau and Vincent Tabard-Cossa. Capture and translocation characteristics of short branched dna labels in solid-state nanopores. *ACS Sensors*, 3(7):1308–1315, 2018. doi: 10.1021/acssensors.8b00165.

Andrew McGregor, Eric Price, and Sofya Vorotnikova. Trace Reconstruction Revisited. In *European Symposium on Algorithms (ESA)*, pages 689–700. Springer, 2014.

Fedor Nazarov and Yuval Peres. Trace reconstruction with $\exp(O(n^{1/3}))$ samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1042–1046, 2017. doi: 10.1145/3055399.3055494. URL http://doi.acm.org/10.1145/3055399.3055494.

Lee Organick, Siena Dumas Ang, Yuan-Jyue Chen, Randolph Lopez, Sergey Yekhanin, Konstantin Makarychev, Miklos Z Racz, Govinda Kamath, Parikshit Gopalan, Bichlien Nguyen, Christopher N Takahashi, Sharon Newman, Hsing-Yeh Parker, Cyrus Rashtchian, Kendall Stewart, Gagan Gupta, Robert Carlson, John Mulligan, Douglas Carmean, Georg Seelig, Luis Ceze, and Karin Strauss. Random access in large-scale DNA data storage. *Nature Biotechnology*, 36:242–248, 2018. URL https://www.nature.com/articles/nbt.4079.

Krishnamurthy Viswanathan and Ram Swaminathan. Improved String Reconstruction Over Insertion-Deletion Channels. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 399–408, 2008.

Kaizhong Zhang and Dennis E. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262, 1989. doi: 10.1137/0218082. URL https://doi.org/10.1137/0218082.

## Appendix A. Missing proofs for $k$-ary trees

**Proof** [Proof of Lemma 13] Being $s$-stable has two conditions. First, we need $G_Y(i) \neq \perp$. Let $G_X^+(i)$ be the union of $G_X(i)$ and the $k$ children of every internal node in $G_X(i)$, where $|G_X^+(i)| = dk + 1$. We will prove that if $Y$ contains $G_X^+(i)$, then $G_Y(i) \neq \perp$, because in fact, $G_Y(i) = G_X(i)$. Since the root is never deleted, all nodes in $G_X^+(i)$ survive in a trace with probability $(1-q)^{dk}$, and so $G_Y(i) = G_X(i)$ with at least this probability.

Assume that $Y$ contains $G_X^+(i)$. Let $G_X(i) = u_0, \ldots, u_{d+k-1}$, and consider building $G_Y(i) = v_0, \ldots, v_{d+k-1}$ using $\pi_i$. We argue recursively: For $t \in [d-1]$, we assume that $v_{t'} = u_{t'}$ for all $t' < t$, and we prove that $v_t = u_t$ as well. The base case $t' = 0$ holds because the root $v_0 = u_0$ is never deleted. Then, since $Y$ contains $G_X^+(i)$, we know that $v_{t'} = u_{t'}$ has exactly $k$ children in $Y$, which are the children of $u_{t'}$ in $X$. Moreover, the left-to-right order of these $k$ children is preserved in the deletion model. Therefore, the child of $v_{t'}$ in position $\pi_i(t')$ must indeed be $u_{t'+1}$ for all $t' < t$. This establishes $v_t = u_t$ for all $t \in \{0, 1, \ldots, d-1\}$. For the leaves of $G_X(i)$, when $v_{d-1} = u_{d-1}$, and $v_{d-1}$ has $k$ children in $Y$, then we must also have $v_d, \ldots, v_{d+k-1} = u_d, \ldots, u_{d+k-1}$.

For the second condition of $s$-stable, consider an internal node $u_t$ in $G_X(i)$ with height $h = d-t$ satisfying $1 \leqslant h \leqslant s$. Let $u'_0, \ldots, u'_{k-1}$ be the children of $u_t$ in $X$. Because $u'_j$ has height $h-1$ in $X$, there is some path with $h$ nodes from $u'_j$ to a leaf in $X$. Consider one such path for each $j = 0, \ldots, k-1$ such that $j \neq \pi_i(t)$. Since there are $k-1$ choices for $j$, let $P_t$ be the union of these $k-1$ paths, where $|P_t| = h(k-1) \leqslant s(k-1)$. The survival of $P_t$ guarantees that $u'_j$ has the correct height for $Y$ to be $s$-stable. Since $|\bigcup_{t=d-s}^{d-1} P_t| \leqslant s^2(k-1)$, and each node survives independently with probability $(1-q)$, we have that $P_{d-s}, \ldots, P_{d-1}$ survive with probability at least $(1-q)^{s^2(k-1)}$.

Combining these two conditions, $Y$ is $s$-stable with probability at least $(1-q)^{dk+s^2k}$. ■

**Proof** [Proof of Lemma 14] Since $Y$ is $s$-stable, $G_Y(i) \neq \perp$. Let $G_Y(i) = v_0, \ldots, v_{d+k-1}$ and $G_X(i) = u_0, \ldots, u_{d+k-1}$, where $v_t$ and $u_t$ have depth $t \in \{0, 1, \ldots, d-1\}$, and $v_{d-1}$ and $u_{d-1}$ have children $v_d, \ldots, v_{d+k-1}$ and $u_d, \ldots, u_{d+k-1}$, respectively. Our strategy is to define an event $\mathcal{E}$ that happens with probability at least $2/3$ and implies that $v_t = u_t$ for $t \leqslant d+k-1$. Consider $t \in [d]$, and let $u'_0, \ldots, u'_{k-1}$ be the children of $u_{t-1}$ in $X$. Define $\mathcal{E}_t$ to be the event that, for every $j \in \{0, 1, \ldots, k-1\}$, at least one node in the subtree rooted at $u'_j$ survives in $Y$. Then, define $\mathcal{E}_{\leqslant m} = \bigcap_{t=1}^m \mathcal{E}_t$ and set $\mathcal{E} = \mathcal{E}_{\leqslant d}$.

We first argue that when $\mathcal{E}_{\leqslant m}$ holds, then $v_t = u_t$ for all $t \leqslant m$. Because the root has not been deleted, we have $v_0 = u_0$. Then, for $t \in [m]$, we assume that $v_{t'} = u_{t'}$ for $t' < t$, and we prove that $v_t = u_t$.

Because $Y$ is $s$-stable, $v_{t-1}$ has $k$ children in $Y$. Denote them $v'_0, \ldots, v'_{k-1}$. We need to show that $u_t$ is in position $\pi_i(t-1)$ among them, so that $v_t = v'_{\pi_i(t-1)} = u_t$. Since $\mathcal{E}_t$ holds, there is some surviving node in $Y$ from the subtree rooted at each original child of $u_{t-1}$ in $X$. Moreover, since $u_{t-1} = v_{t-1}$, this accounts for at least $k$ children of $v_{t-1}$ in $Y$. Because there are exactly $k$ children of $v_{t-1}$, it must be the case that $v'_{\pi_i(t-1)}$ is originally from the subtree rooted at $u_t$ in $X$. In particular, $v'_{\pi_i(t-1)} = u_t$ if and only if $u_t$ survives in $Y$.

We claim that if $u_t$ were deleted, then it would contradict $Y$ being $s$-stable, since we would have $G_Y(i) = \perp$ instead. Indeed, the deletion of $u_t$ would cause $v'_{\pi_i(t-1)}$ to have height less than $d-t$ in $Y$. This would imply that at some depth $d'$ with $t < d' < d$, the node $v_{d'}$ in $G_Y(i)$ would be a leaf, leading to $G_Y(i) = \perp$. We conclude that $u_t$ survives in $Y$, and so that $v_t = v'_{\pi_i(t-1)} = u_t$, as desired.

We have shown that $\mathcal{E}$ guarantees that $v_t = u_t$ for all $t \leqslant d-1$. In particular, $v_{d-1} = u_{d-1}$, and the $k$ children of $v_{d-1}$ in $Y$ must be the children of $u_{d-1}$ in $X$. This finishes the argument that $\mathcal{E}$ implies that $v_t = u_t$ for all $t \leqslant d+k-1$, that is, $G_Y(i) = G_X(i)$.

Now, we prove that $\mathcal{E}$ happens with probability at least $2/3$ in an $s$-stable trace. We prove this in two steps. First, we argue that $\mathcal{E}_{\leqslant d-s}$ occurs with probability at least $2/3$. Then, we show that $\mathcal{E}_{\leqslant d-s}$ implies $\mathcal{E}$. Consider the node $u_{t-1}$ in $G_X(i)$ for $t \in [d-s]$, and let $u'_0, \ldots, u'_{k-1}$ be the $k$ children of $u_{t-1}$ in $X$. Since the height of $u'_j$ is at least $s$, the subtree rooted at $u'_j$ in $X$ contains at least $\sum_{\ell=0}^{s} k^\ell \geqslant k^s$ nodes. The probability that all of these nodes are deleted is at most $q^{k^s}$. Because $s = \left\lceil \log_k \log_{1/q}(3dk) \right\rceil$, this is at most $1/(3dk)$. Taking a union bound over the $k$ children implies that $\mathcal{E}_t$ occurs with probability at least $1 - 1/(3d)$, and taking a union bound over $t \in [d-s]$ implies that $\mathcal{E}_{\leqslant d-s}$ holds with probability at least $2/3$.

The final step is to prove that $\mathcal{E}$ happens with probability one, in an $s$-stable trace, assuming that $\mathcal{E}_{\leqslant d-s}$ holds. More precisely, we will show that $\mathcal{E}_{\leqslant d-s+\ell}$ implies $\mathcal{E}_{d-s+\ell+1}$ for $\ell = 0, 1 \ldots, s-1$. We have already argued that $\mathcal{E}_{\leqslant d-s+\ell}$ guarantees that $v_{d-s+\ell} = u_{d-s+\ell}$. We claim that the $k$ children $v'_0, \ldots, v'_{k-1}$ of $v_{d-s+\ell}$ are the original children of $u_{d-s+\ell}$ in $X$ (and this clearly implies $\mathcal{E}_{d-s+\ell+1}$). Since $Y$ is $s$-stable, there is a path with $s-\ell+1$ nodes from $v'_j$ to a leaf in $Y$. If $v'_j$ were not an original child of $u_{d-s+\ell}$, then all such paths would have at most $s-\ell$ nodes. This implies no children of $u_{d-s+\ell} = v_{d-s+\ell}$ have been deleted in $Y$, and their existence witnesses the survival of the subtrees needed for $\mathcal{E}_{d-s+\ell+1}$. Since this holds for $\ell = 0, 1 \ldots, s$, we conclude that $\mathcal{E} = \mathcal{E}_{\leqslant d}$ follows from $\mathcal{E}_{\leqslant d-s}$ in an $s$-stable trace, and $\mathbf{Pr}[G_Y(i) = G_X(i)] \geqslant \mathbf{Pr}[\mathcal{E}] = \mathbf{Pr}[\mathcal{E}_{\leqslant d-s}] \geqslant 2/3$. ■

## Appendix B. Missing proofs for spiders

**Proof** [Proof of Lemma 15] We index the non-root nodes of the spider according to the DFS ordering described in Section 4. We can uniquely write any $j \in \{0, 1, \ldots, n-1\}$ as $j = d \cdot s_j + r_j$ with $s_j \in \{0, 1, \ldots, n/d - 1\}$ corresponding to a particular path of the spider and $r_j \in \{0, 1, \ldots, d-1\}$ describing where along this path node $j$ is. Consider two nodes, $j = d \cdot s_j + r_j$ and $\ell = d \cdot s_\ell + r_\ell$, with $j \geqslant \ell$. After passing $a$ through the deletion channel to get the trace $b$, $b_\ell$ comes from $a_j$ if and only if $a_j$ is retained, exactly $r_\ell$ of the first $r_j$ nodes in the path of $j$ are retained, and exactly $s_\ell$ of the first $s_j$ paths are retained. This leads to the following generating function:

$$
\begin{aligned}
\mathbb{E}\left[\sum_{\ell=0}^{n-1} b_\ell w^\ell\right] &= (1-q)\sum_{\ell=0}^{n-1} w^\ell \sum_{j=\ell}^{n-1} a_j \binom{r_j}{r_\ell}(1-q)^{r_\ell} q^{r_j - r_\ell}\binom{s_j}{s_\ell} q^{d(s_j - s_\ell)}(1-q^d)^{s_\ell}\mathbf{1}_{\{r_\ell \leqslant r_j\}} \\
&= (1-q)\sum_{j=0}^{n-1} a_j \sum_{\ell=0}^{j} \binom{r_j}{r_\ell}(1-q)^{r_\ell} q^{r_j - r_\ell}\binom{s_j}{s_\ell} q^{d(s_j - s_\ell)}(1-q^d)^{s_\ell} w^\ell \mathbf{1}_{\{r_\ell \leqslant r_j\}} \\
&= (1-q)\sum_{s_j=0}^{n/d-1}\sum_{r_j=0}^{d-1} a_{s_j d + r_j} \sum_{s_\ell=0}^{s_j}\sum_{r_\ell=0}^{r_j} \binom{r_j}{r_\ell}(1-q)^{r_\ell} q^{r_j - r_\ell}\binom{s_j}{s_\ell} q^{d(s_j - s_\ell)}(1-q^d)^{s_\ell} w^{s_\ell d + r_\ell},
\end{aligned}
$$

where we used linearity of expectation and interchanged the order of summation. Observing that the sums are binomial expansions we have that

$$\begin{aligned}\mathbb{E}\left(\sum_{\ell=0}^{n-1} b_\ell w^\ell\right) &= (1-q)\sum_{s_j=0}^{n/d-1}\sum_{r_j=0}^{d-1} a_{ds_j+r_j}(q+(1-q)w)^{r_j}(q^d+(1-q^d)w^d)^{s_j}\\ &= (1-q)\sum_{j=0}^{n-1} a_j(q+(1-q)w)^{j \pmod d}\,(q^d+(1-q^d)w^d)^{\lfloor \frac{j}{d}\rfloor},\end{aligned}$$

which proves the claim. ■

**Proof** [Proof of Lemma 16] Writing $w=\cos(\theta)+i\sin(\theta)$, we see that

$$\begin{aligned}&|(1-q^d)w^d+q^d|^2\\ &\quad= \left|(1-q^d)(\cos(\theta)+i\sin(\theta))^d+q^d\right|^2 = \left|(1-q^d)(\cos(d\theta)+i\sin(d\theta))+q^d\right|^2\\ &\quad= ((1-q^d)\cos(d\theta)+q^d)^2+((1-q^d)\sin(d\theta))^2\\ &\quad= (1-q^d)^2\cos^2(d\theta)+2q^d(1-q^d)\cos(d\theta)+q^{2d}+(1-q^d)^2\sin^2(d\theta)\\ &\quad= (1-q^d)^2+2q^d(1-q^d)\cos(d\theta)+q^{2d} = 1-2q^d+2q^{2d}+2q^d(1-q^d)\cos(d\theta)\\ &\quad= 1-2q^d(1-q^d)(1-\cos(d\theta)).\end{aligned}$$

Now using the fact that $1-\cos(y)\leqslant y^2/2$, as well as the inequality $1-y\geqslant \exp(-4y)$ which holds for all $y\in[0,0.9]$ (in our case indeed $q^d(1-q^d)d^2\theta^2\in[0,0.9]$ for all possible parameter values), we obtain that

$$\left|(1-q^d)w^d+q^d\right|^2 = 1-2q^d(1-q^d)(1-\cos(d\theta)) \geqslant \exp(-4q^d(1-q^d)d^2\theta^2).$$

Taking a square root of the last line shows $|(1-q^d)w^d+q^d|\geqslant \exp(-2q^d(1-q^d)d^2\theta^2)$. Finally, the assumption that $w\in\gamma_L$ implies that $\theta^2\leqslant \pi^2/L^2$ and the claim follows. ■

**Proof** [Proof of Lemma 18] First, we show that $q^d+(1-q^d)|w|^d\leqslant (q+(1-q)|w|)^d$ for all $w\in\mathbb{D}$ and $q\in(0,1)$. This is because

$$\begin{aligned}(q+(1-q)|w|)^d &= \sum_{j=0}^{d}\binom{d}{j}q^j((1-q)|w|)^{d-j} = q^d+\sum_{j=0}^{d-1}\binom{d}{j}q^j((1-q)|w|)^{d-j}\\ &\geqslant q^d+|w|^d\sum_{j=0}^{d-1}\binom{d}{j}q^j(1-q)^{d-j} = q^d+|w|^d(1-q^d),\end{aligned}$$

where we used the inequality $|w|^{-j} \geqslant 1$ which holds when $|w| \leqslant 1$ and $j \geqslant 0$. Combining this inequality with the triangle inequality, we can show the desired upper bound for $\left|\widetilde{A}(w)\right|$:

$$\begin{aligned}
\left|\widetilde{A}(w)\right| &\leqslant \sum_{\ell=\ell^*}^{n-1} |a_\ell| \, |q+(1-q)w|^{\ell \pmod d} \left|q^d+(1-q^d)w^d\right|^{\left\lfloor \frac{\ell}{d}\right\rfloor - \left\lfloor \frac{\ell^*}{d}\right\rfloor} \\
&\leqslant \sum_{\ell=\ell^*}^{n-1} (q+(1-q)\,|w|)^{\ell \pmod d} \left(q^d+(1-q^d)\,|w|^d\right)^{\left\lfloor \frac{\ell}{d}\right\rfloor - \left\lfloor \frac{\ell^*}{d}\right\rfloor} \\
&\leqslant \sum_{\ell=\ell^*}^{n-1} (q+(1-q)\,|w|)^{\ell \pmod d + d\left(\left\lfloor \frac{\ell}{d}\right\rfloor - \left\lfloor \frac{\ell^*}{d}\right\rfloor\right)} \\
&= (q+(1-q)\,|w|)^{-d\left\lfloor \frac{\ell^*}{d}\right\rfloor} \sum_{\ell=\ell^*}^{n-1} (q+(1-q)\,|w|)^{\ell} \\
&\leqslant (q+(1-q)\,|w|)^{-d\left\lfloor \frac{\ell^*}{d}\right\rfloor} \frac{(q+(1-q)\,|w|)^{\ell^*}}{1-(q+(1-q)\,|w|)} \\
&\leqslant \frac{1}{1-(q+(1-q)\,|w|)} = \frac{1}{(1-q)(1-|w|)},
\end{aligned}$$

where we used that $q+(1-q)|w| < 1$ and $\ell^* - d\lfloor \ell^*/d \rfloor \geqslant 0$. Note that the same upper bound holds for $|A(w)|$ as well, since $|A(w)| \leqslant \left|\widetilde{A}(w)\right|$ for all $w \in \mathbb{D}$. ■