

Reconstructing Arbitrary Trees from Traces in the Tree Edit Distance Model

Thomas Maranzatto

Department of Mathematics, Statistics, and Computer Science,
University of Illinois at Chicago

January 14, 2022

Abstract

In this paper, we consider the problem of reconstructing trees from traces in the tree edit distance model. Previous work by Davies et al. [10] analyzed special cases of reconstructing labeled trees. In this work, we significantly expand our understanding of this problem by giving general results in the case of arbitrary trees. Namely, we give: a reduction from the tree trace reconstruction problem to the more classical string reconstruction problem when the tree topology is known, a lower bound for learning arbitrary tree topologies, and a general algorithm for learning the topology of any tree using techniques of Nazarov and Peres [19]. We conclude by discussing why arbitrary trees require exponentially many samples under the left propagation model.

1 Introduction

In this paper, we study the complexity of reconstructing trees from traces under the tree edit distance model, as defined by Davies et al. [10]. *Tree trace reconstruction* is a generalization of the traditional *trace reconstruction* problem, where we observe noisy samples of a binary string after passing it through a deletion channel several times [1]. Trace reconstruction and tree trace reconstruction have a variety of applications including in archival DNA storage, sensor networks, and linguistic reconstruction [2, 9, 14, 3].

1.1 String trace reconstruction

Below we define the string trace reconstruction problem.

Definition 1 (string trace reconstruction). *In the string trace reconstruction problem, a trace of a string s on n binary bits is obtained by running the string through a deletion channel that removes each bit of s independently with*

probability q . Each trace is generated independently of the other traces. The goal is to reconstruct s with probability at least $1 - \delta$ using as few traces as possible. Denote by $T(n, q, \delta)$ the minimum number of traces needed to reconstruct a string with n bits with deletion rate q and success probability $1 - \delta$. We sometimes hide the dependency on q by writing $T(n, \delta)$ when q is known.

This problem has been studied extensively, yet there is still an exponential gap in the upper and lower bounds on trace sample complexity. Chase [6] recently proved that $\exp(\tilde{O}(n^{1/5}))$ traces are sufficient to reconstruct a string, improving on the previous bound of $\exp(O(n^{1/3}))$ traces that was independently discovered by Nazarov and Peres [19] and De et al. [12].

Many variants on trace reconstruction have been considered. In the average-case setting, the unknown input string is chosen uniformly at random, and improved upper bounds are known in this case [1, 15, 16, 17]. In particular, Holden et al. [16] showed that $O(\exp(C \log^{1/3} n))$ traces are sufficient to recover the string.

Other settings which yield improved bounds are in the smoothed-analysis setting where the unknown string is perturbed before generating traces [7], in the coded setting where the string is confined to a pre-determined set [4, 8, 21], and in the approximate setting where the goal is to output a string that is close to the original in edit-distance [11].

While theoretically exciting, there is also a direct application of trace reconstruction to DNA data storage where algorithms are deployed to recover the data from noisy samples [2, 9, 20]. Clearly there is a practical need to reduce the sample complexity as this impacts the time and cost of retrieving archival data stored with DNA. For example, [9] were able to store a 53,426 word book, 11 JPG images, and a JavaScript program (a total of 5.27 megabits) in a DNA medium over 9 years ago. Finding efficient reconstruction algorithms are crucial to the scalability of this storage medium.

1.2 Tree trace reconstruction

Recent work has allowed the creation of robust branching DNA structures for storage [14, 18]. This naturally leads to the tree trace reconstruction problem posed by Davies et al. [10]. In their model, a tree of known topology has n nodes, and each node v also carries one hidden bit of information $\ell(v) \in \{0, 1\}$. This corresponds to each position in a string carrying one bit of information. They gave reductions from tree trace reconstruction to string trace reconstruction in three specific cases.

Definition 2 (tree trace reconstruction). *In the tree trace reconstruction problem, a trace of a tree T on n nodes is obtained by removing each node of T independently with probability q . Each trace is generated independently of the other traces. The goal is to reconstruct T with probability at least $1 - \delta$ using as few traces as possible.*

The notion of removing a node from T is ambiguous, and there are two main models that have been studied by Davies et al. [10]. In this paper we focus

primarily on the *tree edit distance* (TED) deletion model: when a node is deleted, all of its children become children of its parent. The deleted node's children take its place as a contiguous subsequence in the left-to-right order. Another model we briefly consider is called *left-propagation*. There, when a node is deleted, recursively replace every node (together with its label) in the left-only path starting at the node 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.

The tree trace reconstruction problem trivially generalizes the string reconstruction problems; for example, a path graph is a tree, and under node deletions is equivalent to the string reconstruction problem. Some tree structures simplify the problem; for example, Davies et al. [10] showed significant improvements over the $O(\exp(Cn^{1/3}))$ bound for the cases of k -ary trees and spider graphs. However, it was *a priori* possible that some tree structures may present harder problems and require even more samples than their string counterparts.

In addition to the interesting theoretical questions this model raises, it also has applications to real-world problems in constructing branched DNA structures. This topic is relevant for long-term data structure storage and progress on it could help to improve data density in the storage medium.

For example, He et al. [14] were able to synthesize short-armed star DNA structures with 3 to 12 arms in total. Their process allowed for control over how many arms are produced, and they were able to detect the number of arms in a given sample. Furthermore, Karau and Tabard-Cossa [18] investigated T and Y shaped DNA strands, where the backbone of the molecule is twice as long as the branches. They provide biochemical methods for robustly detecting these structures. Creating tree trace reconstruction algorithms is then crucial for the viability of branched DNA as an archival storage option.

1.3 Our results

In this paper we give the following results.

- In section 3, we give a reduction from the problem of reconstructing trees from traces to the problem of reconstructing strings from traces, in the case where the tree structure is known. Our reduction only increases the running time by a multiplicative factor of n .¹
- In Section 4, we give a lower bound for learning the topology of a tree from tree traces. In particular, we construct a “linked list” tree with n nodes and simulate a string reconstruction problem by adding leaves below each of these nodes, either to the left or to the right of the main chain. Learning traces from this tree is very similar to learning string traces.
- In Section 5, We give an algorithm for learning traces using $\exp(\tilde{O}(n^{1/5}))$ samples. There, we use the argument from Chase [5], but applied to our flattened tree traces. This bound is independent of learning the tree labels.

¹This answers Question 2 in Section 6.1 of Davies et al. [10].

- In Section 6, we show that $O((1-q)^{-n})$ samples are needed to reconstruct a tree whose traces are generated from the left-propagation deletion channel.

2 Related Work

The problem of string-trace reconstruction has received considerable attention since it was first introduced by [1]. Despite this there is still an exponential gap in the necessary number of traces and the sufficient number of traces needed to reconstruct an arbitrary input string [5, 6, 12, 15, 19]. The best lower bound thus far is due to Chase [5], where he showed that $\tilde{\Omega}(\log^{5/2} n)$ traces are necessary, and the best upper bound is due to Chase [6], where the same author showed that $\exp(\tilde{O}(n^{1/5}))$ traces are sufficient. Previous to this, Nazarov and Peres [19] and De et al. [12] independently achieved the upper bound of $\exp(O(n^{1/3}))$ traces.

Variants of trace reconstruction have been proposed in order to develop new techniques to close the exponential gap in the original problem formulation. Davies et al. [10] investigated a new problem of reconstructing trees from their traces. The authors investigated two classes of trees; complete k -ary trees and spider graphs. They also introduced the “tree edit distance” (TED) and “left-propagation” deletion channels.

For a complete k -ary tree T the authors considered the general case for arbitrary k , as well as when k is ‘large’ compared to the total number of nodes. Under TED, if $k \geq O(\log^2(n))$, then a complete k -ary tree can be reconstructed using $\exp(O(\log_k n)) \cdot T(k, 1/n^2)$ traces generated from T . With no restrictions on k , $\exp(O(k \log_k n))$ traces suffice to reconstruct T . Under left-propagation, if $k \geq O(\log n)$ then $T(O(\log_k n + k), 1/n^2)$ traces suffice to reconstruct T . For arbitrary k then $O(n^\gamma \log n)$ traces suffice, where $\gamma = \ln\left(\frac{1}{1-q}\right)\left(\frac{c'k}{\ln n} + \frac{1}{\ln k}\right)$. The authors uses primarily combinatorial arguments to achieve these bounds, which is in contrast to the current methods used in string trace reconstruction. It was given as an open problem to extend these combinatorial arguments to general trees.

The case of spider graphs was broken into small leg-depth and large leg-depth. Note that for spider graphs, TED and left-propagation are identical deletion channels. For depth $d \leq \log_{1/q} n$ and $q > 0.7$, then $\exp(O(d(nq^d)^{1/3}))$ traces suffice to reconstruct the spider. For depth $d \geq \log_{1/q} n$ and all $q > 1$, then $2T(d, 1/2n^2)$ traces suffice to reconstruct the spider. The algorithms proposed here generalize the mean-based methods of De et al. [12] and Nazarov and Peres [19] to multiple independent strings, where some strings have a chance of being completely removed.

3 Learning trees of known topology

In this section, we show that it is indeed not the case that tree reconstruction can require more traces than the corresponding string reconstruction. In particular, in the case that the tree structure is known, we reduce the problem of learning

trees from traces with no overhead in the number of traces required and only a multiplicative linear overhead in the time complexity. We show this by giving a traversal of the tree that converts the tree to a string in linear time such that node deletions occur in their corresponding positions in the string. This allows string trace reconstruction algorithms to be used for tree trace reconstruction.

We start by giving the main lemma of this section, which (informally) shows that the order of nodes visited by a pre-order traversal of a tree is preserved under deletions. We note that a pre-order traversal is usually defined for binary trees; we define a pre-order traversal of general trees as recursively visiting the root first and then all children left to right.

Lemma 3. *Let T be a tree and T' be a tree obtained from T via an arbitrary set of deletions under either the trace edit distance or the left propagation model. If u, v are any two nodes in T' such that a pre-order traversal of T' visits u before v then a pre-order traversal of T will also visit u before visiting v .*

Proof. First, we consider the tree edit distance model. Given the deletion of an arbitrary node w , we consider how it effects the traversal order of the remaining nodes. Figure 1 shows the general case when a node w is removed, with the subtrees numbered in the order they are visited according to a pre-order traversal.

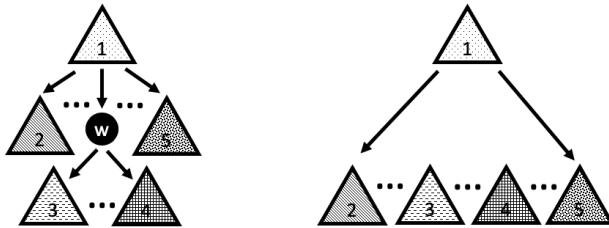


Figure 1: The generic picture before (left) and after (right) node w is removed. The subtrees are labeled in the order they are visited in a pre-order traversal. Note that on the left, w is visited immediately before the tree labeled with the number 3.

If u and v fall in the same subtree, their visitation order is clearly not affected. Otherwise, if u is in a tree that's visited earlier than when the tree of v is visited on the left of Figure 1, then it is also visited earlier on the right. This simple illustration captures all of the cases (i.e. pairs of trees where u and v can occur), which finishes the first part of the proof. \square

Let T' be a trace of T and let $v_1 \dots v_{n'}$ be the nodes of T' indexed by the order they are visited in a pre-order traversal. Let

$$s(T') = \ell(v_1)\ell(v_2) \dots \ell(v_{n'}).$$

Let T'_1, T'_2, \dots, T'_m be traces of T . This procedure allows us to obtain m strings

$$s(T'_1), s(T'_2), \dots, s(T'_m).$$

Using Lemma 3, if m is sufficiently large, we can run a string trace reconstruction algorithm on this sequence of strings m to obtain the string $s(T)$, which allows us to label T 's nodes in the order given by a pre-order traversal.

Hence, we can conclude the following.

Theorem 4. *Let T be any tree whose topology is known, its labels can be reconstructed using $T(n, q, \delta)$ traces in the worst case.*

Using the reduction above and applying the results of Chase [6] for recovering any string from $\tilde{O}(\exp(n^{1/5}))$ traces, we get the following.

Corollary 5. *Let T be any tree whose topology is known, its labels can be reconstructed using $\tilde{O}(\exp(n^{1/5}))$ traces in the worst case.*

Similarly, applying the known results of Hartung et al. [13] for recovering a random string from $O(\exp(C \log^{1/3} n))$ traces, we obtain the following as a corollary.

Corollary 6. *Let T be any tree whose topology is known, if T 's nodes are labeled uniformly at random, its labels can be reconstructed using $O(\exp(C \log^{1/3} n))$ traces.*

A natural open question is what happens in trees whose topology is not known. In that case, it makes sense to attempt to recover the topology.

4 A lower bound on learning the topology

One may then ask about the problem of reconstructing the topology of an unknown tree. For this problem, we can assume all the labels are the same, as to give the least possible information to the learner. We call such a tree an unlabeled tree.

For the lower bound we show there exists an unlabelled tree that naturally encodes the string trace reconstruction problem. Given a string S we construct the tree T_S in two stages. First, create a path tree with $|S| + 2\ell$ vertices, where $\ell = \Theta\left(\frac{\ln(1/\delta) + \ln(T(n, \delta))}{\ln(1/q)}\right)$. Next, for each bit $S_i \in S$, if $S_i = 0$ add a left child to the node in position $(\ell + i)$ in the path. Otherwise $S_i = 1$ so add a right child to node $(\ell + i)$ in the path. Call the subtree which includes left and right leaves the **encoding** of S . We now show that this family of trees is sufficiently hard to learn under the TED model.

Lemma 7. *Let T_S be a tree constructed from a string S using the process defined above. Under the TED deletion channel T_S requires $\Omega(T(n, q^2, \delta))$ traces for reconstruction.*

Proof. By the Hoeffding bound, we can recover the longest path in T_S with probability $1 - \delta$ using $O(n \log(1/\delta))$ samples. This gives the ‘backbone’ of the string encoding. Choosing

$$\ell = \Theta\left(\frac{\ln(1/\delta) + \ln(T(n, \delta))}{\ln(1/q)}\right)$$

ensures that with probability $1 - \delta$ in all $T(n, \delta)$ independent trials, at least one of the vertices survive in both the upper and lower ‘buffer sections’ of the tree. We know which positions in the backbone the leaves will occur in T_S , thus it suffices to learn the orientation of the leaves with high probability.

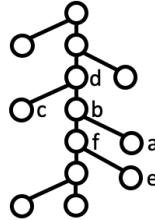


Figure 2: Example configuration of a leaf node a and the leaves nearest it. The survival of a is dependent on the nodes in its 3-neighborhood.

For this proof, we refer to Figure 2 which shows a possible configuration of the main path and leaves oriented on either side. The exact orientations of the leaves do not matter in the following analysis. Say node a is completely removed in a trace if both it and its parent are deleted from the T_S . The probability that a is completely removed in a trace is dependent on its parent being removed as well. In general there may be many ways leaves could be arranged in a trace if they are not completely deleted. Suppose there existed some oracle that given a trace generated from T_S could replace ‘backbone’ nodes and leaf nodes that were only partially deleted. Could we hope to reconstruct the original tree with fewer traces using this oracle? Here, nodes are removed with probability q^2 . Thus even an algorithm endowed with this oracle cannot place these leaves in the tree, so leaves survive with probability $1 - q^2$.

There is a 1-to-1 correspondence between leaf position in the tree and bit value in a string. Further, the survival of a bit with the oracle learner is $O(p^2)$, so the final algorithm would require $\Omega(T(n, q^2, \delta))$ traces to reconstruct T_S . This is the best any algorithm could hope to do, as nodes that are completely removed by definition do not leave any information in the trace. \square

This gives a natural lower bound to the tree trace reconstruction problem, as we have no guarantee on the input tree to the problem.

Theorem 8. *For any two trees T_R and T_S encoding binary strings R and S respectively, then $\Omega(T(n, q^2, \delta))$ traces are required to distinguish between T_R and T_S .*

Using the above argument along with the results of Chase [6], it follows T_S can be reconstructed using $\tilde{O}(\exp(n^{1/5}))$ traces.

Corollary 9. *Let T_S be a tree encoding of binary string S . Then for any fixed deletion probability q , $\tilde{O}(\exp(n^{1/5}))$ traces suffice to reconstruct T_S with probability $1 - \delta$.*

5 Recovering fuzzy trees of degree $\tilde{O}(n^{1/5})$

For any tree, call a leaf terminal if all of its siblings are leaves. We define a *fuzzy tree of degree m* as a tree where all terminal leaves have $m - 1$ siblings.

Recall the result from [5] that gives $T(n, \delta) \leq \exp(\tilde{O}(n^{1/5}))$.

Theorem 10. *Given a fuzzy tree A of degree $\tilde{O}(n^{1/5})$ and probability δ , we can reconstruct A using $\tilde{O}(n^{1/5})$ traces with high probability.*

Proof. Sample $T(n, \delta)$ traces from A . For trace m , construct two binary strings S_0^m, S_1^m from a preorder traversal of the trace. S_0^m uses the alphabet $\{0, 2\}$ and S_1^m uses the alphabet $\{2, 1\}$. S_1^m is constructed by writing a 1 when the traversal moves down an edge, and a 2 when the traversal encounters a leaf. S_2^m is constructed by writing a 0 when the traversal moves up an edge, and a 2 when the traversal encounters a leaf.

Because of the assumption that A is fuzzy, for every trace the probability that there exists a node with all children deleted is less than $1/(n \cdot \exp(\tilde{O}(n^{1/5})))$. By the union bound the probability that any trace contains a node where all children are deleted is less than $1/n$.

A has two unknown strings S_0 and S_1 constructed analogously to the trace strings above. It is easy to see that a node deletion in A corresponds to a single bit deletion in S_0 and S_1 . This property holds for multiple deletions as well. Further with high probability no node will have all its children removed. Because only one edge is represented by each bit in the two strings, only one edge is removed in a TED deletion, and all children of a leaf are not deleted with high probability, we conclude that with high probability deletions in S_0 and S_1 behave as *i.i.d* deletions in the string deletion channel model. Therefore, by applying the results from [5] we obtain the desired result. \square

6 Recovery under Left-Propagation

Here we show that learning under left propagation requires exponentially many samples. Namely we show the existence of two simple trees A and B that can only be distinguished if the left propagation deletion model removes no nodes. This happens with probability $(1 - q)^n$, so the worst-case sample bound would then be $\Omega((1 - q)^{-n})$. This bound is also what any nontrivial algorithm should try to overcome.

We call $LP_k(A)$ the k 'th order left propagation trace set of tree A , which is the set of all traces obtainable by removing k nodes from A . The trace set contains no probabilistic information, it simply categorizes what traces could be generated regardless of how unlikely they are to appear. We approach this by showing the trace sets of A and B under left-prop are identical for any non-zero number of deletions. We use the shorthand A_n and B_n to show that A and B both have n non-root nodes.

Theorem 11. *For all $n > 3$ and all $m \leq n$ there exist unlabelled trees A and B containing n non-root nodes each, such that $LP_m(A) = LP_m(B)$.*

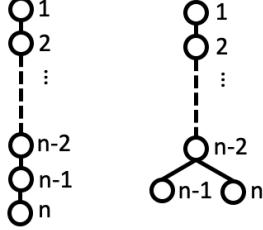


Figure 3: The two trees we are investigating. Notice that the trees' structures are identical until node $n - 2$. Removing any single node in either tree results in the tree A_{n-1} .

Proof. For $n > 3$, consider the unlabelled tree A with n nodes, where every internal node has exactly one child, including the root. Also consider the tree B with n nodes which is constructed by taking A_{n-1} and adding a sibling to the single leaf.

The set $LP_1(A_n)$ only contains one item, namely $A_{(n-1)}$. There is no dependence on a deletion probability here, we are requiring that one and only one node is removed from the tree. Set $m < n$. Using the above as a base case, by recursion we see that $LP_m(A_n) = A_{(n-m)}$.

An analogous argument holds for $LP_1(B_n)$. If any node in the chain is deleted, then the left leaf is promoted to be the parent of the right leaf. This structure is a long chain with $n - 1$ non-root nodes, the definition of $A_{(n-1)}$. If either leaf is removed, then we obviously get a chain of $n - 1$ non-root nodes. Thus $LP_1(B_n) = A_{(n-1)}$. Because of this the same recursive argument for $LP_m(A_n)$, implying $LP_m(B_n) = A_{(n-m)}$. So we have

$$\begin{aligned} LP_m(A_n) &= LP_m(B_n) \\ &= A_{(n-m)} \end{aligned}$$

finishing the proof. \square

An easy corollary follows from above.

Corollary 12. *In the worst case, $\Omega((1 - q)^{-n})$ traces are required to learn a tree topology in the Left Propagation model.*

7 Discussion

In this paper we gave many connections between the string trace reconstruction and tree trace reconstruction problems. Our results exploited these connections and string trace reconstruction techniques to give state-of-the-art bounds for reconstructing arbitrary trees from traces.

References

- [1] Tugkan Batu, Sampath Kannan, Sanjeev Khanna, and Andrew McGregor. Reconstructing strings from random traces. In *SODA*, pages 910–918. SIAM, 2004.
- [2] Vinnu Bhardwaj, Pavel A. Pevzner, Cyrus Rashtchian, and Yana Safonova. Trace reconstruction problems in computational biology. *IEEE Transactions on Information Theory*, page 1–1, 2020.
- [3] Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths, and Dan Klein. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*, 110(11):4224–4229, March 2013. Publisher: National Academy of Sciences Section: Physical Sciences.
- [4] Joshua Brakensiek, Ray Li, and Bruce Spang. Coded trace reconstruction in a constant number of traces, 2019.
- [5] Zachary Chase. New lower bounds for trace reconstruction, 2019.
- [6] Zachary Chase. New upper bounds for trace reconstruction, 2020.
- [7] Xi Chen, Anindya De, Chin Ho Lee, Rocco A. Servedio, and Sandip Sinha. Polynomial-time trace reconstruction in the smoothed complexity model, 2020.
- [8] M. Cheraghchi, R. Gabrys, O. Milenkovic, and J. Ribeiro. Coded trace reconstruction. *IEEE Transactions on Information Theory*, 66(10):6084–6103, 2020.
- [9] George M. Church, Yuan Gao, and Sriram Kosuri. Next-generation digital information storage in dna. *Science*, 337(6102):1628–1628, 2012.
- [10] Sami Davies, Miklós Z. Rácz, and Cyrus Rashtchian. Reconstructing trees from traces. In Alina Beygelzimer and Daniel Hsu, editors, *COLT*, volume 99, pages 961–978. PMLR, 2019.
- [11] Sami Davies, Miklos Z. Racz, Cyrus Rashtchian, and Benjamin G. Schiffer. Approximate trace reconstruction, 2020.
- [12] Anindya De, Ryan O’Donnell, and Rocco A Servedio. Optimal mean-based algorithms for trace reconstruction. *Annals of Applied Probability*, 29(2):851–874, 2019.
- [13] Lisa Hartung, Nina Holden, and Yuval Peres. Trace reconstruction with varying deletion probabilities. In *ANALCO 2018*, pages 54–61. SIAM, 2018.
- [14] Liquun He, Philipp Karau, and Vincent Tabard-Cossa. Fast capture and multiplexed detection of short multi-arm dna stars in solid-state nanopores. *Nanoscale*, 11:16342–16350, 2019.

- [15] Nina Holden and Russell Lyons. Lower bounds for trace reconstruction, 2018.
- [16] Nina Holden, Robin Pemantle, and Yuval Peres. Subpolynomial trace reconstruction for random strings and arbitrary deletion probability. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pages 1799–1840. PMLR, 06–09 Jul 2018.
- [17] Thomas Holenstein, Michael Mitzenmacher, Rina Panigrahy, and Udi Wieder. Trace reconstruction with constant deletion probability and related results. In *SODA*, 2008.
- [18] Philipp 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. PMID: 29874054.
- [19] Fedor Nazarov and Yuval Peres. Trace reconstruction with $\exp(O(n^{1/3}))$ samples. In *STOC*, pages 1042–1046. ACM, 2017.
- [20] 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(3):242–248, March 2018.
- [21] Sundara Rajan Srinivasavaradhan, Michelle Du, Suhas Diggavi, and Christina Fragouli. On maximum likelihood reconstruction over multiple deletion channels. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 436–440. IEEE, 2018.