Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 167 (2016)

Simplified Data Structure Lower Bounds for Dynamic Graph Connectivity

Sivaramakrishnan Natarajan Ramamoorthy * Anup Rao†

May 20, 2017

Abstract

The problem of dynamic connectivity in graphs has been extensively studied in the cell probe model. The task is to design a data structure that supports addition of edges and checks connectivity between arbitrary pair of vertices. Let w, t_q, t_u denote the word size, expected query time and worst case update time of a data structure for connectivity on graphs of size n. We provide simplified proofs of the following results:

- Any data structure for connectivity with error at most $\frac{1}{32}$ must have $t_q \geq \Omega\left(\frac{\log n}{\log w t_u}\right)$. This was proved in the landmark paper of Fredman and Saks [FS89].
- For every $\delta > 0$ and data structure for connectivity that makes no errors, if $t_u = o\left(\frac{\log n}{\log \log n}\right)$, then $t_q \geq \Omega\left(\delta n^{1-2\delta}\right)$. This was proved by Patrascu and Thorrup in [PT11].

1 Introduction

Dynamic data structures play a key role in algorithm design. Dijkstra's algorithm, Fibonacci heaps, hashing, Van Emde Boas trees are some classical examples of dynamic data structures. Proving lower bounds on the performance of dynamic data structures is usually challenging. Fredman and Saks [FS89], in their landmark paper, were the first to establish tight lower bounds for several dynamic problems. Since then, several works have developed techniques to prove lower bounds on various dynamic data structure problems [PD06, Păt07, PT11, Lar12, Yu16, WY16].

A *dynamic data structure* is an algorithmic primitive for efficiently maintaining some data. In this paper, we consider the classical problem of graph connectivity. We wish to store a graph, on n vertices, allowing for the addition of edges and be able to ask whether or not two given vertices are connected. We use the standard cell-probe model of Yao [Yao81]. The data structure consists of a collection of memory cells that are used to store the data (the graph), together with an efficient algorithm for performing updates and queries on the data.

The data structure is said to have update time t_u, query time t_q and word size w if the algorithm that maintains the graph uses a table of cells, each of size w and supports two kinds of operations:

*Computer Science and Engineering, University of Washington, sivanr@cs.washington.edu. Supported by the National Science Foundation under agreement CCF-1420268 and CCF-1524251

†Computer Science and Engineering, University of Washington, anuprao@cs.washington.edu. Supported by the National Science Foundation under agreement CCF-1420268 and CCF-1524251.

ISSN 1433-8092

- Updates: these are operations that change the graph being stored. In our case we require that the data structure support the operation $\mathsf{Add}(u, v)$, which adds an edge between the vertices u, v in the graph. During each such update, the data structure reads and writes to at most t_u of the cells.

- Queries: these are operations that compute some function of the data. In our case, we require that the data structure support the query $\mathsf{Conn}(u, v)$ which outputs 1 if u, v are connected in the graph, and 0 otherwise. During each such operation, the data structure is allowed to read at most t_q of the cells.

The above definitions can be extended to a setting where the algorithm has access to randomness and is allowed to makes errors. We say that the data structure is an ϵ-error data structure if the probability the data structure computes a query incorrectly is at most ϵ. Here the probability is over the random choices of the data structure algorithm.

A popular variant of the connectivity problem is the Union-Find problem. Every component in the graph is now associated with a representative. The updates correspond to adding an edge between the representative of two components and the query corresponds to finding the representative of the component that the given vertex belongs to. Tarjan in [Tar75] showed that $n-1$ updates and m queries can be executed in $O(n+m\alpha(m,n))$ operations, where α is the inverse Ackerman function. The classic UNION-FIND algorithm gives $O(1)$ updates and $O(\log n)$ query time in the worst case. Blum in [Blu86] designed a data structure that supports queries in worst case time $O(\log n/\log t_u)$. This trade-off lets you achieve $t_u = t_q = O(\log n/\log\log n)$. The same data structure supports operations for dynamic connectivity with time complexity $O(\log n/\log\log n)$.

In this paper, we focus our attention on lower bounds for dynamic connectivity. Fredman and Saks [FS89] in their celebrated result were the first to prove a lower bound of $t_q = \Omega\left(\frac{\log n}{\log w t_u}\right)$ in the cell probe model. When $w = O(\log n)$, this matches the upper bound given by [Blu86]. The trade-off was later improved in [ABAR99] to $t_q = \Omega\left(\frac{\log n}{\log t_u}\right)$, for $w = O(\log n)$. This remained the best trade-off for any dynamic problem, until Patrascu and Demaine in [PD06] showed $\mathsf{min}\{t_u, t_q\} = \Omega(\log n)$, for $w = O(\log n)$, for PARTIAL-SUM and fully dynamic connectivity (for the problem of fully dynamic connectivity, the updates also allow deletion of edges).

It is natural to ask if these lower bounds are tight when $t_u = o(\log n/\log\log n)$ or $t_q = o(\log n/\log\log n)$. In this context, Patrascu and Thorrup in [PT11] showed that for every $\delta > 0$, if $w = O(\log n)$ and $t_u = o(\log n/\log\log n)$, $t_q = \Omega\left(\delta n^{1-2\delta}\right)$. Their lower bound applies to zero error randomized data structures.

In this paper, we revisit both the lower bound questions discussed above. We simplify the proofs of [FS89] and [PT11]. Our results are summarized in the following subsection.

1.1 Our Results

For the rest of the discussion, we take t_q to be the expected query time and t_u to be the worst case update time. We prove the following two theorems,

Theorem 1 ([FS89])**.** *For any $\frac{1}{32}$- error data structure for dynamic connectivity, it holds that*

$$t_q \geq \Omega\left(\frac{\log n}{\log w t_u}\right).$$

Our proof starts with the chronogram approach of [FS89], and deviates from theirs in a crucial step. For a parameter $B = (wt_u)^2$, we sample a random forest consisting of two B-ary trees, one rooted at the vertex 1, and the second rooted at the vertex 2. We pick a sequence of updates to add the edges of this graph to the data structure, such that the edges closest to the leaves are added first. We then query the connectivity of a random pair of leaves in the graph.

The choice of the graph ensures that the number of edges at a particular level L of the trees is a factor B larger than the number of edges that will be added in future updates. In the analysis, we fix all the updates that correspond to edges that are not at level L. We label each leaf of the tree with either 1 or 2 depending on which tree it belongs to. We show that even after fixing the updates outside level L, the entropy of these labels is close to the number of vertices at level L.

B is chosen to ensure that all reads performed by the algorithm after the edges of L have been added are not enough to recover all the information about the edges of L. This is because the the number of updates after level L is about a factor of B smaller than the number of edges in L, and each update can generate at most t_u reads.

This lets us argue that even after fixing the cells written during future updates, the entropy of the labels on the leaves remains quite high. A simple application of Shearer's lemma from information theory is then used to show that this means that the algorithm must touch at least one cell that was modified during the updates corresponding to L to answer the connectivity query. Overall, this argument lower bounds the query time by the depth of the tree, as desired.

Theorem 2 ([PT11])**.** *For $\delta > 0$, any 0-error data structure for dynamic connectivity must have,*

$$t_q \geq \frac{n^{1-2\delta} \cdot \log_B n}{\log n + \log t_u + w} \cdot \Omega\left(\delta \log n - t_u\left(\log\log n + \frac{\log n + \log t_u + w}{\log_B n}\right)\right),$$

where $B = (wt_u)^2$.

Corollary 3. *For $\delta > 0$, any 0-error data structure for dynamic connectivity with $t_u = o\left(\frac{\log n}{\log\log n}\right), w = O(\log n)$ must have $t_q = \Omega\left(\delta n^{1-2\delta}\right)$.*

To prove Theorem 2, we use the distribution described in [PT11]. Here we sample n^δ uniformly random trees, each of the same depth. The root of each tree has approximately $n^{1-\delta}$ children, and all other vertices in the tree have B children.

As before, we construct a sequence of updates, where edges closer to the leaves are added first, and label every leaf by the name of its root. We argue, as before, that on fixing the updates on all but one level L, and cells corresponding to the future updates, the entropy of the labels remains about $|L| \log n^\delta > n^{1-\delta} \log n^\delta$. An application of Shearer's lemma shows that this means that the labels of about $n^{1-\delta}$ leaves will have roughly the same entropy.

Next we use an encoding argument to show that the data structure can be used to encode these labels using approximately $w \cdot (n^{1-\delta} t_u + n^\delta t_q)$ bits. The argument works by executing an additional $n^{1-\delta}$ updates and n^δ connectivity queries and storing the contents of all the cells that were touched during the process. We argue that the contents of this carefully chosen set of cells allows one to reconstruct the labels of the leaves. Our lower bound is finally obtained by comparing the entropy of the labels with the length of the encoding.

2 Preliminaries

Unless otherwise stated, logarithms in this article are computed base two. Random variables are denoted by capital letters and values they attain are denoted by lower-case letters. For example, A may be a random variable and then a denotes a value A may attain and we may consider the event $A = a$. The expected value of a real valued random variable A is denoted by $\mathbb{E}[A]$. Given $a = a_1, a_2, \ldots, a_n$, we write $a_{\leq i}$ to denote $a_1, \ldots, a_i$. We define $a_{>i}$ and $a_{\leq i}$ similarly. We write a_{-i} to denote $a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_n$ and $[\ell]$ denotes the set $\{1, 2, \ldots, \ell\}$.

We often work with subsets of cells written by the data structure and sequence of updates. When C is a subset of the cells, C denotes the contents and memory locations of C, and $|C|$ as the number of cells in C. Similarly, when U is a sequence of updates, $|U|$ denotes the number of updates in U, and U denotes the description of every update with the update order.

The *entropy* of a discrete random variable A, is defined to be

$$\mathsf{H}(A) = \sum_a \Pr[A = a] \cdot \log \frac{1}{\Pr[A = a]}.$$

For two random variables A, B, the entropy of A conditioned on B is defined to be

$$\mathsf{H}(A|B) = \sum_{a,b} \Pr[A = a, B = b] \cdot \log \frac{1}{\Pr[A = a|B = b]}.$$

The *binary entropy function* is defined to be

$$\mathsf{h}(p) = p \log \frac{1}{p} + (1 - p) \log \frac{1}{1 - p},$$

for $0 \leq p \leq 1$. The entropy satisfies the chain rule:

Proposition 4 (Chain Rule)**.** $\mathsf{H}(A_1A_2|B) = \mathsf{H}(A_1|B) + \mathsf{H}(A_2|BA_1)$.

As consequence of the chain rule, demonstrated in [Rad01], we get the following lemma:

Lemma 5 (Shearer's Lemma)**.** *Let X be a random variable on $\{0,1\}^n$ and S be a distribution on $2^{[n]}$ such that $\Pr[i \in S] \geq \mu$, then*

$$\mathop{\mathbb{E}}_{S}[\mathsf{H}(X_S)] \geq \mu \mathsf{H}(X),$$

where X_S denotes the projection $X_{i_1} \cdots X_{i_k}$ for $S = \{i_1, \cdots, i_k\}$.

Proposition 6. *Let A and B be random variables on $[n]$, R be a random variable independent of A and f be a mapping such that $\Pr[f(B, R) = A] = 1$. Then, $\mathsf{H}(A) \leq \mathsf{H}(B)$.*

Proof. By the chain rule for entropy,

$$\begin{aligned} \mathsf{H}(A) = \mathsf{H}(A|R) &\leq H(AB|R) \\ &= \mathsf{H}(B|R) + \mathsf{H}(A|BR) \\ &\leq \mathsf{H}(B). \end{aligned}$$

We used the facts that $\mathsf{H}(A|RB) = 0$, since R and B determine A. □

Proposition 7. *For every integer valued random variable A, $\mathsf{H}(A) \leq O(\log \mathbb{E}[|A|])$.*

Proposition 8. *For every random variable A with an injective binary encoding of expected length l, $\mathsf{H}(A) \leq l + O\left(\sqrt{l}\right)$.*

The following proposition gives an algorithm to test membership in a set.

Proposition 9. *For every $0 < p < 1$, there exists a random variable D such that for every $S \subseteq [m]$, there is a random variables D' and an algorithm A such that*

1. *for every $x \in S$, $\mathsf{A}(D, D', x) = 1$, and for every $x \notin S$, $\mathsf{A}(D, D', x) = 1$ with probability p and 0 otherwise.*
2. *$\mathsf{H}(D') \leq O\left(|S| \cdot \log \frac{1}{p}\right)$*

Proof. Let $D = T_1, T_2, \cdots$ be an infinite sequence of sets such that for every i, T_i is a random subset of $[m]$ in which every element of $[m]$ is included in T_i with probability p. Let D' be the random variable denoting the integer l such that $S \subseteq T_l$ and for $l' < l$, $S \nsubseteq T_{l'}$.

We now compute the expected value of D'. For every i, the probability that $S \subseteq T_i$ is $p^{|S|}$. Therefore, the expected value of D' is $\frac{1}{p^{|S|}}$. This is because, the expected value of D' satisfies the following equation:

$$\mathbb{E}\left[D'\right] = \Pr[D' = 1] \cdot 1 + \Pr[D' \neq 1] \cdot (1 + \mathbb{E}\left[D'\right]).$$

Proposition 7 implies that $\mathsf{H}(D') \leq O\left(|S| \cdot \log \frac{1}{p}\right)$.

We now describe the algorithm that tests membership in S. A on input $x \in [m]$ outputs 1 if $x \in T_{D'}$ and 0 otherwise. By the definition of $T_{D'}$, for every $x \in S$, the algorithm always outputs 1. For $x \notin S$, the probability that $x \in T_{D'}$ is p. This completes the the proof that the algorithm satisfies property 1 of the proposition. □

Proposition 10 (Stirling's Approximation)**.** *For $n \in \mathbb{N}$,*

$$\log \binom{2n}{n} \geq 2n - 1 - \log \sqrt{n}$$

and

$$\log n! = n \log n - n + O(\log n)$$

Proposition 11. *For $n \geq 1$, and $0 < x \leq 1$, $(1-x)^n \leq 1 - xn + n^2x^2$.*

Proof. Consider $f(x) = (1-x)^n - 1 + xn - n^2x^2$. We have,

$$f'(x) = -n(1-x)^{n-1} + n - 2n^2x = n(1 - (1-x)^{n-1} - 2nx).$$

Since $(1-x)^n \geq 1 - xn$, we note that,

$$\begin{aligned}(1-x)^{n-1} + 2nx &> 1 - x(n-1) + 2nx \\ &= 1 + nx + x > 1.\end{aligned}$$

This implies that $f'(x) < 0$. Therefore, $f(x) \leq f(0) = 0$ for $0 \leq x \leq 1$, as required. □

3 Dynamic Connectivity

Without loss of generality, we can take $t_q \leq n^2$, where n is the number of vertices in the underlying graph. This is because, one can maintain the indicator vector of the edge set using $\binom{n}{2}$ cells, and can answer connectivity queries in $\binom{n}{2}$ time by reading the indicator vector and execute the updates in time 1 by modifying the bit corresponding to the edge inserted in the indicator vector.

Before we prove Theorem 1 and Theorem 2, we establish a few facts about entropy of a subset of cells written by the data structure.

Proposition 12. *Let C be a subset of the cells and s be the number of cells. Then, $\mathsf{H}(C) \leq |C|(\log s + w)$.*

Proof. To encode the set C, it is sufficient to write down a list of memory locations of cells in C along with their contents. It requires $\log \binom{s}{|C|} \leq |C| \log s$ bits to write down the locations of C. Since each cell is w bits long, it requires $w|C|$ bits to write down the contents of C. This amounts to a total of $|C|(\log s + w)$ bits to encode C. □

Proposition 13. *Let $U = U_1, U_2$ be a sequence of updates to the data structure with l updates in U_2, and C be the set of cells accessed while executing the updates in U_2, then $\mathsf{H}(C|U_2) \leq w \cdot t_u \cdot l$.*

Proof. Let J be the number of cells accessed while executing U_2. Let $D_1, D_2, \cdots, D_k$, for $k = l \cdot t_u$, be such that D_i denotes the contents of the i'th cell accessed while executing the updates U_2. In addition, for every $J < j' \leq k$, $D_{j'} = D_J$. Note that the location of the i'th cell accessed during the execution of updates U_2 may not be determined by U_2. We now proceed to show that $D_1, D_2, \cdots, D_k$, and U_2 determine C. For $1 \leq i \leq J$, $D_{<i}$ along with U_2 determines the location of the i'th cell accessed and D_i determines its new contents before the $i+1$'th access. Therefore, C is determined by U_2 and $D_1, D_2, \cdots, D_k$. In other words, $\mathsf{H}(C|D_1, \cdots, D_k, U_2) = 0$. We now have,

$$\begin{aligned}
\mathsf{H}(C|U_2) &\leq \mathsf{H}(C, D_1, \cdots, D_k|U_2) \\
&= \mathsf{H}(D_1, D_2, \cdots, D_k|U_2) + \mathsf{H}(C|U_2, D_1, \cdots, D_k) \\
&\leq w \cdot t_u \cdot l,
\end{aligned}$$

as required. □

For the sake of simplicity, in the discussion to follow, we set $w = 2 \log n$, where n is the number of vertices in the input graph.

3.1 Proof of Theorem 1

We first present a hard distribution on a sequence of updates and queries. Set $B = (t_u \log n)^2$ and $d = \log_B \frac{n}{2}$. The underlying graph will be a forest consisting of 2 B-ary trees rooted at vertices 1 and 2 with $n/2$ leaves each (see Figure 2). The set of instructions to the data structure is the sequence UQ sampled in Figure 1, where $U = U_0 \cdots U_{d-1}$. We now analyze the behavior of any data structure on this input sequence.

Let $\{l_1, l_2, \cdots, l_n\}$ be the set of vertices at depth d. Define $X = (X_1, \cdots, X_n)$ such that $X_i = \mathsf{Conn}(l_i, 1)$. For $k \in \{0, 1, \cdots, d-1\}$, define C_k to be the random variable denoting the set of cells written during the execution of the k'th batch of updates, and not overwritten during the execution of the future updates $U_{>k}$. Let $p_k = \Pr_{UQ}[Q \text{ reads a cell from } C_k]$. The following lemma implies the lower bound on t_q as stated in Theorem 1, since $t_q \geq \sum_{k=0}^{d-1} p_k$.

- Sample two random $B-$ary trees rooted at vertices 1 and 2 with $n/2$ leaves each.

The depth of a vertex v is the distance from the vertex in $\{1, 2\}$ that is connected to v.

- For $k \in \{0, 1, \cdots, d-1\}$, let U_k denote the k'th batch of updates. The updates in U_k correspond to adding the edges between vertices at depth $d-k$ and $d-k-1$ in some arbitrary order.
- Let Q denote the random variable for the connectivity query between a random pair of vertices which are at depth d.

Figure 1: Hard Distribution for Dynamic Connectivity with parameters B, n, d.

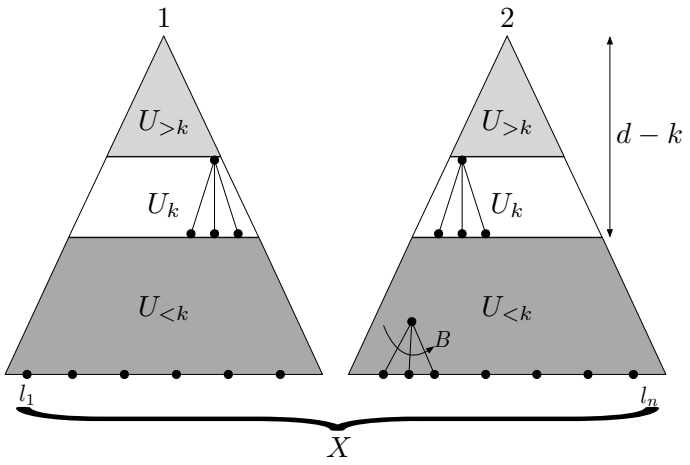

Figure 2: The input graph

Lemma 14. *$p_k \geq \frac{1}{16} - o(1)$ for all $k \in \{0, \cdots d-2\}$.*

We prove the above lemma at the end of this subsection.

Lemma 15. *For every $k \in \{0, 1, \cdots, d-2\}$, $\mathsf{H}(X|C_{>k}U_{-k}) \geq \frac{n}{B^k} \cdot (1 - o(1))$.*

Proof. By the chain rule for entropy,

$$\begin{aligned}\mathsf{H}(X|C_{>k}U_{-k}) &\geq \mathsf{H}(X|U_{-k}) - \mathsf{H}(C_{>k}|U_{-k}) \\ &\geq \mathsf{H}(X|U_{-k}) - \mathsf{H}(C_{>k}|U_{>k}),\end{aligned}$$

where the inequality $\mathsf{H}(C_{>k}|U_{>k}) \geq \mathsf{H}(C_{>k}|U_{-k})$ follows from the fact that conditioning only decreases entropy.

We first show that $\mathsf{H}(X|U_{-k}) \geq \frac{n}{B^k} - 1 - \log\sqrt{\frac{n}{B^k}}$. The input distribution dictates that $X|U_{-k}$ is a balanced n/B^k bit vector, drawn uniformly at random. Therefore $\mathsf{H}(X|U_{-k}) = \log\binom{2t}{t} \geq 2t - 1 - \log\sqrt{t}$, where $2t = n/B^k$ and the inequality follows from Proposition 10.

An upper bound on $\mathsf{H}(C_{>k}|U_{>k}) = o(n/B^k)$ would imply the lemma. We have,

$$|U_{>k}| = \sum_{j=k+1}^{d} \frac{n}{B^j} \leq \frac{n}{B^{k+1}} \cdot \frac{B}{B-1}.$$

Proposition 13 and the upper bound on $|U_{>k}|$ imply,

$$\mathsf{H}(C_{>k}|U_{>k}) \leq \frac{n}{B^{k+1}} \cdot \frac{B}{B-1} \cdot t_u \cdot 2\log 2n \leq o\left(\frac{n}{B^k}\right),$$

which follows from the choice of B. □

Lemma 16. *Let $k \in \{0, 1, \cdots, d-2\}$ and S be a random subset of $[n]$ of size 2. Then,*

$$\mathop{\mathbb{E}}_{S}\left[\mathsf{H}(X_S|C_{>k}U_{-k})\right] \geq 2 - o(1).$$

Proof. $X|U_{-k}$ denotes the connectivity labels of an equivalence class of size n/B^k, in which 2 leaves belong to the same class if their parents at depth $d-k$ are the same. The projection to S corresponds to a projection onto the equivalence classes given by the sets of parents of elements in S at depth $d-k$.

The distribution on the projection $S' \subseteq [n/B^k]$ has the property that $\Pr[i \in S'] = 1 - \binom{n-B^k}{2}/\binom{n}{2}$. We now proceed to lower bound this probability.

$$\binom{n-B^k}{2}/\binom{n}{2} = \frac{(n-B^k)}{n} \cdot \frac{n-B^k-1}{n-1} \leq \left(1 - \frac{B^k}{n}\right)^2 = 1 - \frac{2B^k}{n} + \frac{B^{2k}}{n^2}.$$

Therefore $\Pr[i \in S'] \geq \frac{2B^k}{n} - \frac{B^{2k}}{n^2}$. By applying Shearer's lemma with the projection being S', we get

$$\mathop{\mathbb{E}}_{S}\left[\mathsf{H}(X_S|C_{>k}U_{-k})\right] \geq \left(\frac{2B^k}{n} - \frac{B^{2k}}{n^2}\right) \cdot \mathsf{H}(X|C_{>k}U_{-k}).$$

The bound on $\mathsf{H}(X|C_{>k}U_{-k})$ from Lemma 15 and the fact that $B^k/n = o(1)$ completes the proof. □

Lemma 17. *For every $k \in \{0, 1, \cdots, d-2\}$, $S \subset [n]$ with $|S| = 2$,*

$$\mathsf{H}(X_S|C_{>k}U_{-k}) \leq \frac{9}{8} + 2p_k + \mathsf{h}(p_k) + \mathsf{h}\left(\frac{1}{16}\right).$$

Proof. Let

$$E = \begin{cases} 1, & \text{if } Q \text{ reads a cell from } C_k \\ 0, & \text{otherwise} \end{cases}$$

Let $S = \{i, j\}$ and E' be the indicator variable for the error in the output of $\mathsf{Conn}(i, j) = X_i \oplus X_j$. By the subadditivity of entropy and the chain rule for entropy, we have

$$\begin{aligned}
\mathsf{H}(X_S|C_{>k}U_{-k}) &\leq \mathsf{H}(X_S EE'|C_{>k}U_{-k}) \\
&\leq \mathsf{H}(EE'|C_{>k}U_{-k}) + \mathsf{H}(X_S|C_{>k}U_{-k}, EE') \\
&\leq \mathsf{H}(E) + \mathsf{H}(E') + \mathsf{H}(X_S|C_{>k}U_{-k}, EE') \\
&\leq \mathsf{h}(p_k) + \mathsf{h}\left(\frac{1}{16}\right) + \mathsf{H}(X_S|C_{>k}U_{-k}, EE'),
\end{aligned}$$

- Sample $n^{1-\delta}$ complete $B-ary$ trees rooted at $r_1, \cdots, r_{n^{1-\delta}}$, each with n leaves.
- Let $v_1, \cdots, v_{n^\delta}$ be n^δ vertices (disjoint from the vertices of the $B-ary$ trees) and edges incident to them are $\left\{(v_i, r_j) \middle| (i-1)\cdot n^{1-2\delta} \leq j < i \cdot n^{1-2\delta}, i \in [n^\delta]\right\}$.

The depth of a vertex v is the distance from the vertex in $\{v_1, \cdots, v_{n^\delta}\}$ that is connected to v. The sequence of updates are as follows,

- For $k \in [d]$, let U_k denote the k'th batch of updates which corresponds to adding edges between vertices at depth $d-k+1, d-k+2$ in some arbitrary order.
- U_{d+1} corresponds to the addition of edges that are incident to $v_1, \cdots, v_{n^\delta}$.

Figure 3: Hard Distribution used in Theorem 2 with parameters B, d, n, δ

where the last inequality follows from the facts that $\Pr[E=1] = p_k$ and $\Pr[E'=1] \leq \frac{1}{16}$.

Now observe that $\mathsf{H}(X_S | C_{>k} U_{-k}, E=0, E'=0) \leq 1$, since $E=0$ and $E'=0$ imply that the algorithm knows the output $\mathsf{Conn}(i,j)$ without reading a cell from C_k. Since X_S is supported on a universe of size 2, we get that

$$\mathsf{H}(X_S | C_{>k} U_{-k}, EE') \leq 1 + 2 \cdot \Pr[E=1 \text{ or } E'=1] \leq 2p_k + 9/8.$$

This implies that $\mathsf{H}(X_S | C_{>k} U_{-k}) \leq \mathsf{h}(p_k) + \mathsf{h}\left(\frac{1}{16}\right) + 2p_k + \frac{9}{8}$. □

Proof of Lemma 14. Lemma 16 implies that for every $k \in \{0, \cdots, d-2\}$,

$$\mathop{\mathbb{E}}_{S \subseteq [n], |S|=2} [\mathsf{H}(X_S | C_{>k} U_{-k})] \geq 2 - o(1).$$

Putting together the fact that $h\left(\frac{1}{16}\right) + \frac{1}{16} < 0.5$ and the conclusion of Lemma 17, we get that $h(p_k) + 2p_k \geq 1/2 - o(1)$, for every $k \in \{0, \cdots, d-2\}$. It follows that $p_k > 1/16 - o(1)$, which completes the proof of Lemma 14. □

3.2 Threshold Phenomenon in Dynamic Connectivity

In this subsection we prove Theorem 2 using the hard distribution defined in [PT11]. Set $B = (t_u \log n)^2$ and $d = \log_B n$. Execute the sequence of updates $U_1, U_2, \cdots, U_{d+1}$ which are sampled as in Figure 3.

Let $\{l_1, \cdots, l_{n^{2-\delta}}\}$ be the set of vertices at depth $d+1$. Define $T(l_i) \in$ to be the random variable denoting the vertex in $\{v_1, \cdots, v_{n^\delta}\}$ that is connected to l_i, and $L = (T(l_1), \cdots, T(l_{n^{2-\delta}}))$. Let $S \subseteq [n^{2-\delta}]$ be a random subset of size $n^{1-\delta}$, and C^* be the set of cells read/written during the executing of the following operations: $\forall j \in S$, execute $\mathsf{Add}(l_j, T(l_j))$. Then, $\forall j' \in [n^\delta - 1]$ execute $\mathsf{Conn}(v_{j'}, v_{j'+1})$, and if the output is 0, execute $\mathsf{Add}(v_{j'}, v_{j'+1})$. Note that the total number of operations executed is at most n^2.

We now discuss a technique to restrict the space of the data structure to polynomial in n, t_u, while incurring no additional error to the data structure. Let s be the space of the data structure. Consider an infinite sequence $\mathcal{G} = g_1, g_2, \cdots$ of random hash functions $g_i : [s] \to \left[n^8 (t_u + n^4)^2\right]$.

Given g_i, consider the following simulation of the data structure: Every access to cell j of the original data structure is simulated by an access to $g_i(j)$. We have, for every $j, k \in [s]$ such that $\Pr[g_i(j) = g_i(k)] = \frac{1}{n^8(t_u+n^4)^2}$. Since, the total number of operations executed is at most n^2 and $t_q \leq n^4$, the total number of cells accessed is at most $n^2(t_u + n^4)$. Therefore, by the union bound, the probability that two distinct cells hash to the same number is at most $\frac{1}{n^2}$.

Let Z be the random variable denoting the smallest integer i such that the simulation of the data structure using g_i has zero collisions. We have, $\mathbb{E}[Z] = \left(1 - \frac{1}{n^2}\right)^{-1} \leq 2$. By Proposition 7, $\mathsf{H}(Z) \leq O(1)$. From now onward, we shall simulate the given data structure with g_Z and any access to cell j of the data structure is an access to the cell $g_Z(j)$.

Observation 18. $\mathbb{E}[|C^*|] \leq 2n^{1-\delta}t_u + n^\delta t_q$.

Proof. The inequality follows from the fact that the total number of $\mathsf{Add}(,)$ operations executed is at most $2n^{1-\delta}$ and the total number of $\mathsf{Conn}(,)$ queries answered is n^δ. □

Lemma 19. $\mathbb{E}[|C^*|] \geq \frac{n^{1-\delta}\log_B n}{\log n + \log t_u} \cdot \Omega\left(\delta \log n - t_u \log\log n\right)$.

Proof. Let $C_1, \cdots, C_{d+1}$ be subsets of cells such that C_k is the set of cells written while executing updates in U_k and not overwritten while executing the future updates $U_{>k}$. We now invoke Proposition 9 (the parameter p is to be chosen later) to test membership of cells within $C_1, \cdots, C_{d+1}$. Let p and $\mathcal{R}_1, Y_1, \cdots, Y_{d+1}$ be the real number and the random variables given by Proposition 9[1], in which $\forall k \in [d+1]$, $\mathcal{R}_1, Y_k$ is used by the algorithm to test membership in C_k. For the rest of the proof, we assume that $k \leq (1-\delta)\log_B n$.

Claim 20. $\mathsf{H}(L|U_{-k}) \geq \frac{\delta n^{2-\delta}\log n}{B^k} \cdot (1 - o(1))$.

Proof. The total number of ways of partitioning a set of size $t = n^{2-\delta}/B^k$ into n^δ sets of equal size is $\frac{t!}{((t/n^\delta)!)^{n^\delta}}$. Since L is an uniform distribution on the support of the partition, we have

$$\begin{aligned}
\mathsf{H}(L|U_{-k}) = \log\left(\frac{t!}{((t/n^\delta)!)^{n^\delta}}\right) &= \log t! - n^\delta \cdot \log(t/n^\delta)! \\
&= t\log t - t + O(\log t) - n^\delta\left(\frac{t}{n^\delta}\log\frac{t}{n^\delta} - \frac{t}{n^\delta} + O\left(\log\frac{t}{n^\delta}\right)\right) \\
&\geq t\log n^\delta - n^\delta \cdot O\left(\log\frac{t}{n^\delta}\right),
\end{aligned}$$

where the third equality followed from Proposition 10. Now $n^\delta \cdot O\left(\log\frac{t}{n^\delta}\right) = o(t\log n^\delta)$ by the choice of the upper bound on k. □

Lemma 21. $\mathsf{H}(L|U_{-k}C_{>k}Y_kZ) \geq \frac{\delta n^{2-\delta}\log n}{B^k} \cdot \left(1 - O\left(\frac{t_u \log\frac{1}{p}}{\delta\log n}\right) - o(1)\right)$.

Proof. By the chain rule for entropy, we have,

$$\begin{aligned}
\mathsf{H}(L|U_{-k}C_{>k}Y_kZ) &\geq \mathsf{H}(L|U_{-k}) - \mathsf{H}(C_{>k}Y_kZ|U_{-k}) \\
&\geq \mathsf{H}(L|U_{-k}) - \mathsf{H}(C_{>k}|U_{>k}) - \mathsf{H}(Y_k) - \mathsf{H}(Z),
\end{aligned}$$

[1]We remark that the random variables $Y_k, \cdots, Y_{d+1}$ play the role of the certificates given by Bloom filters in [PT11].

where we used the fact that conditioning only decreases entropy. We have,

$$\begin{aligned}|U_{>k}| &= \sum_{j=k+1}^{d} \frac{n^{2-\delta}}{B^j} \\ &\leq \frac{n}{B^{k+1}} \cdot \frac{B}{B-1}.\end{aligned}$$

Proposition 13 implies,

$$\mathsf{H}(C_{>k}|U_{>k}) \leq \frac{n^{2-\delta}}{B^{k+1}} \cdot \frac{B}{B-1} \cdot t_u \cdot 2\log 2n^{2-\delta} \leq o\left(\frac{n^{2-\delta}}{B^k}\right),$$

which follows from the choice of B. In addition, $|C_k|$ is at most $|U_k|t_u = \frac{n^{2-\delta}}{B^k} \cdot t_u$. By Proposition 9, $H(Y_k) \leq O\left(\frac{n^{2-\delta}t_u \log \frac{1}{p}}{B^k}\right)$. Also, $\mathsf{H}(Z) \leq O(1)$. The above bounds and Claim 20 imply the lemma. □

Lemma 22. $\mathsf{H}(L_S|SU_{-k}C_{>k}Y_kZ) \geq \delta \cdot n^{1-\delta} \cdot \log n \cdot \left(1 - O\left(\frac{t_u \log \frac{1}{p}}{\delta \log n}\right) - o(1)\right)$.

Proof. Let L^k denote the set of nodes at depth $d-k+2$ in the graph. $\mathsf{H}(L_S|SU_{-k}C_kY_kZ)$ is the same as $\mathsf{H}(L^k_{S'}|S'U_{-k}C_kY_kZ)$, in which S' is the collection of parents of elements in S at depth $d-k+2$. The distribution on the projection $S' \subseteq [n^{2-\delta}/B^k]$ has the property that $\Pr[i \in S'] = 1 - \binom{n^{2-\delta}-B^k}{n^{1-\delta}}/\binom{n^{2-\delta}}{n^{1-\delta}}$. We now lower bound this probability.

$$\begin{aligned}\binom{n^{2-\delta}-B^k}{n^{1-\delta}}/\binom{n^{2-\delta}}{n^{1-\delta}} &= \prod_{j=0}^{B^k-1}\left(\frac{n^{2-\delta}-n^\delta-j}{n^{2-\delta}-j}\right) \\ &\leq \left(1-\frac{1}{n}\right)^{B^k} \\ &\leq 1 - \frac{B^k}{n} + \frac{B^{2k}}{n^2},\end{aligned}$$

where the last inequality follows from Proposition 11. Therefore $\Pr[i \in S'] \geq \frac{B^k}{n} - \frac{B^{2k}}{n^2}$. Applying Shearer's lemma gives,

$$H(L_S|SU_{-k}C_{>k}Y_kZ) \geq \left(\frac{B^k}{n} - \frac{B^{2k}}{n^2}\right) \cdot \frac{\delta}{B^k} \cdot \left(n^{2-\delta}\log n - O\left(\frac{n^{2-\delta}t_u \log \frac{1}{p}}{\delta}\right) - o(n^{2-\delta}\log n)\right),$$

as desired. We used the fact that the upper bound on k implies $B^k/n = o(1)$. □

For every $k \leq (1-\delta)\log_B n$, let $C^*_k = C_k \cap C^*$. By definition, $\mathbb{E}\left[|C^*|\right] = \sum_{k=1}^{d} \mathbb{E}\left[|C^*_k|\right]$.

Lemma 23. $\mathsf{H}(L_S|S, U_{-k}, C_{>k}Y_kZ) \leq \mathbb{E}\left[|C^*_k|\right] \cdot O(\log n + \log t_u) + p \cdot \mathbb{E}\left[|C^*|\right] \cdot O(\log n + \log t_u)$.

The Verification Procedure

Input: $a_1, \cdots, a_{n^{1-\delta}}, C_k^*, Y$.
Operations to execute: Let $s_1 < s_2 < \cdots < s_{n^{1-\delta}}$, such that $s_i \in S$. For every $j \in [n^{1-\delta}]$, execute $\mathsf{Add}(l_{s_j}, a_j)$. Then, $\forall j' \in [n^\delta - 1]$ query $\mathsf{Conn}(v_{j'}, v_{j'+1})$ and when the output is 0, execute $\mathsf{Add}(v_{j'}, v_{j'+1})$.
Cell Access: We now describe how to simulate an access to a cell c.

1. If $c \in C_{>k}$, return the contents of c.
2. If $c \notin C_{>k}$, check if $c \in C_k$ using Y_k and the algorithm from Proposition 9.
3. If the algorithm indicates that $c \in C_k$, then check whether $c \in C_k^*$ or $c \in Y$.
   (a) If $c \in C_k^*$, return its contents.
   (b) If $c \in Y$, return its contents.
   (c) if $c \notin C_k^* \cup Y$, ABORT.
4. If the algorithm indicates that $c \notin C_k$, then $c \in C_{<k}$.The contents of cells in $C_{<k}$ is consistent with its contents after the execution of $U_{<k}$, which the data structure knows.

If all the connectivity queries output 0, then return SUCCESS. Return a FAILURE if at least one query outputs 1 or a cell access returns an ABORT.

Figure 4: Simulating the data structure algorithm

Proof. Fix $S, U_{-k}, C_{>k}, Y_k, Z$. Let $a_1, \cdots, a_{n^{1-\delta}} \in [n^\delta]$ and $\mathcal{R}_2$ be the randomness used by the data structure. Let Y be the set of cells in $C_{<k}$ which are reported to be in C_k by the algorithm given by Proposition 9, while executing the following operations: For every $j \in [n^{1-\delta}]$, execute $\mathsf{Add}(l_j, T(l_j))$. Then, for every $j' \in [n^\delta - 1]$, query $\mathsf{Conn}(v_{j'}, v_{j'+1})$ and when the output is 0, execute $\mathsf{Add}(v_{j'}, v_{j'+1})$.

We claim that there exists a mapping f such that $\Pr[f(C_k^*, Y, \mathcal{R}_1, \mathcal{R}_2, \mathcal{G}) = L_S] = 1$. We first describe the mapping f before discussing the implications of this claim. The goal is prove that $T(l_1), \cdots, T(l_{n^{1-\delta}})$ can be recovered using C_k^* and Y. For every $a_1, \cdots, a_{n^{1-\delta}} \in [n^\delta]$, execute the verification procedure described in Figure 4. Since the data structure makes no errors, the verification procedure returns a SUCCESS only when $(a_1, a_2, \cdots, a_{n^{1-\delta}}) = L_S$. By the guarantee of the data structure, in expectation over $\mathcal{R}_1, \mathcal{R}_2$, we can recover L_S with probability 1. This establishes the existence of a mapping f with the desired properties.

Proposition 6 implies that

$$\mathsf{H}(L_S | S, U_{-k}, C_{>k}, Y_k Z) \leq \mathsf{H}(C_k^*) + \mathsf{H}(Y).$$

We now upper bound $\mathsf{H}(C_k^*)$ and $\mathsf{H}(Y)$. To encode C_k^*, we have to specify its memory locations and contents. By Proposition 12 and the definition of g_Z, $\mathsf{H}(C_k^*) \leq \mathbb{E}[|C_k^*|] \cdot O(\log n + \log t_u)$. We now upper bound $\mathsf{H}(Y)$. Proposition 8 and the fact that $\mathbb{E}[|Y|] \leq p \cdot \mathbb{E}[|C^*|]$ imply that $\mathsf{H}(Y) \leq p \cdot \mathbb{E}[|C^*|] \cdot O(\log n + \log t_u)$. This completes the proof of Lemma 23. □

We now proceed to complete the proof of Lemma 19. Lemma 22 and Lemma 23 imply,

$$\mathbb{E}\left[|C_k^*|\right] \geq \Omega\left(\frac{\delta n^{1-\delta}\log n}{\log n + \log t_u} - p\,\mathbb{E}\left[|C^*|\right] - \frac{n^{1-\delta}t_u \log \frac{1}{p}}{\log n + \log t_u}\right).$$

Either $\mathbb{E}\left[C^*\right] = \Omega\left(\delta n^{1-\delta}\log_B n\right)$ or not. In the former case, we are done. In the later case, setting $p = \frac{1}{\log^2 n}$ implies that

$$\mathbb{E}\left[|C_k^*|\right] \geq \frac{n^{1-\delta}}{\log n + \log t_u} \cdot \Omega\left(\delta \log n - t_u \log\log n\right).$$

Since the above inequality holds for every $k \leq (1-\delta)\log_B n$, we get,

$$\mathbb{E}\left[|C^*|\right] \geq \frac{n^{1-\delta}\log_B n}{\log n + \log t_u} \cdot \Omega\left(\delta \log n - t_u \log\log n\right),$$

which is the desired bound on $\mathbb{E}\left[|C^*|\right]$. This completes the proof of Lemma 19. □

We return to the proof of Theorem 2. The conclusions of Lemma 19 and Observation 18 imply,

$$n^{2\delta-1}t_q \geq \frac{\log_B n}{\log n + \log t_u} \cdot \Omega\left(\delta \log n - t_u\left(\log\log n + \frac{\log n + \log t_u}{\log_B n}\right)\right).$$

This completes the proof of Theorem 2.

References

[ABAR99] Alstrup, Ben-Amram, and Rauhe. Worst-case and amortised optimality in union-find (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing (STOC'99)*, New York, May 1999. Association for Computing Machinery.

[Blu86] N. Blum. On the single-operation worst-case time complexity of the disjoint set union problem. *SIAM J. Comput.*, 15(4):1021–1024, 1986.

[FS89] M. Fredman and M. Saks. The cell probe complexity of dynamic data structures. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 1989.

[Lar12] K. G. Larsen. The cell probe complexity of dynamic range counting. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 85–94, 2012.

[Păt07] M. Pătraşcu. Lower bounds for 2-dimensional range counting. In *Proc. 39th ACM Symposium on Theory of Computing (STOC)*, pages 40–46, 2007.

[PD06] M. Pătraşcu and E. D. Demaine. Logarithmic lower bounds in the cell-probe model. *SIAM Journal on Computing*, 35(4):932–963, 2006. See also STOC'04, SODA'04.

[PT11] M. Pătraşcu and M. Thorup. Don't rush into a union: Take time to find your roots. In *Proc. 43rd ACM Symposium on Theory of Computing (STOC)*, pages 559–568, 2011. See also arXiv:1102.1783.

[Rad01] J. Radhakrishnan. Entropy and counting. *IIT Kharagpur, Golden Jubilee Volume, on Computational Mathematics, Modelling and Algorithms (Ed. JC Mishra),*, 2001.

[Tar75] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *J. ACM*, 22(2):215–225, Apr. 1975.

[WY16] O. Weinstein and H. Yu. Amortized dynamic cell-probe lower bounds from four-party communication. *Electronic Colloquium on Computational Complexity (ECCC)*, 23, 2016.

[Yao81] A. Yao. Should tables be sorted? *JACM: Journal of the ACM*, 28, 1981.

[Yu16] H. Yu. Cell-probe lower bounds for dynamic problems via a new communication model. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 362–374, 2016.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il