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

New Randomized Data Structure Lower Bounds for Dynamic Graph Connectivity

Sivaramakrishnan Natarajan Ramamoorthy * Anup Rao†

November 1, 2016

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 cell width, expected query time and worst case update time of a data structure for connectivity on graphs of size n. We prove the following,

- For every $\delta > 0$ and any data structure for connectivity with error at most $\frac{1}{4n^\delta}$, if $t_u = o\left(\frac{\log n}{\log\log n}\right)$, then $t_q \geq \Omega\left(\delta n^{1-3\delta}\right)$. Patrascu and Thorrup in [PT11] show the same for data structures with zero errors.

In addition, we simplify the proof of dynamic connectivity lower bound established in the landmark paper of Fredman and Saks[FS89]. The result states that for any data structure for connectivity with constant error bounds, $t_q \geq \Omega\left(\frac{\log n}{\log(w+\log n)t_u}\right)$.

*Computer Science and Engineering, University of Washington, sivanr@cs.washington.edu. Supported by the National Science Foundation under agreement CCF-1420268 and CCF-1016509

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

ISSN 1433-8092

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, allowing for the addition 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:

- 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 the given vertex belong 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 show a lower bound of $t_q = \Omega\left(\frac{\log n}{\log wt_u}\right)$ 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 $t_q = \Omega\left(\frac{\log n}{\log t_u}\right)$. 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 PARTIAL-SUM and fully dynamic connectivity(for the problem of fully dynamic connectivity, the updates allow deletion of edges too).

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] show that for dynamic connectivity if $t_u = o(\log n/\log\log n)$, $t_q = \Omega\left(\delta n^{1-2\delta}\right)$ for every $\delta > 0$. Their lower bound applies to zero error randomized data structures, but don't extend to ones with errors.

In this paper, we revisit both the lower bound questions discussed above. We reprove the result of [FS89], and show the trade-off demonstrated in [PT11] for polynomially small error randomized data structures. Our results can be summarized as follows,

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*

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

Our proof starts with the chronogram approach of [FS89], and deviates from thiers 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. *For $\delta > 0$, any $\frac{1}{4n^\delta}$-error data structure for dynamic connectivity obeys,*

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

Corollary 3. *For $\delta > 0$, any $\frac{1}{4n^\delta}$-error data structure for dynamic connectivity with $t_u = o\left(\frac{\log n}{\log\log n}\right), w = O(\log n)$ has $t_q = \Omega\left(\delta n^{1-2\delta}\right)$.*

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 performing 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.

[PT11] use the same set of cells implicitly in their argument, but they reduce to a 2-party communication game computing the equality function. This does not yield them strong lower bounds when there are errors, since one can compute the equality function quite efficiently when errors are allowed.

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 $A = a$. 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. Similarly, we write a_{-i} to denote $a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_n$. $[\ell]$ denotes the set $\{1, 2, \ldots, \ell\}$.

We use the notation $p(a)$ to denote both the distribution on the variable a, and the number $\Pr_p[A = a]$. The meaning will be clear from context. We write $p(a|b)$ to denote either the distribution of A conditioned on the event $B = b$, or the number $\Pr[A = a|B = b]$. Again, the meaning will be clear from context. Given a distribution $p(a, b, c, d)$, we write $p(a, b, c)$ to denote the marginal distribution on the variables a, b, c (or the corresponding probability). We often write $p(ab)$ instead of $p(a, b)$ for conciseness of notation. If W is an event, we write $p(W)$ to denote its probability according to p. We denote by $\mathbb{E}_{p(a)}[g(a)]$ the expected value of $g(a)$ with respect to a distributed according to p.

We often work with subset of cells written by the data structure and sequence of updates. When C is a subset of cells, we think of C as a description of its contents along with the memory locations and $|C|$ as the number of cells. Similarly, when U is a sequence of updates, we think of $|U|$ as the number of updates and U to be the description of every update along with the order.

The *entropy* of a random variable A, conditioned on B is defined to be

$$\mathsf{H}_p(A|B) = \sum_{a,b} p(ab) \log \frac{1}{p(a|b)}$$

For a binary random variable A, we denote the entropy of A to be

$$\mathsf{h}(p(0)) = -\left[p(0) \log p(0) + (1 - p(0)) \log(1 - p(0))\right],$$

where $p(0) = \Pr_p(A = 0)$ 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 $\Pr[f(B,R) = A] \geq 1 - \delta$. Then, $\mathsf{H}(A) \leq 1 + \mathsf{H}(B) + \delta \log n$.*

Proof. Let E be the indicator random variable for $f(B,R) = A$. By the chain rule of entropy,

$$\begin{aligned}
\mathsf{H}(A) = \mathsf{H}(A|R) &\leq H(AB|R) \\
&= \mathsf{H}(B|R) + \mathsf{H}(A|B,R) \\
&\leq \mathsf{H}(B) + \mathsf{H}(AE|RB) \\
&= \mathsf{H}(B) + \mathsf{H}(E|RB) + \mathsf{H}(A|ERB) \\
&\leq \mathsf{H}(B) + 1 + \Pr[E=0]\mathsf{H}(A|RB, E=0) \\
&\leq 1 + \mathsf{H}(B) + \delta \log n.
\end{aligned}$$

We used the facts that $\mathsf{H}(E|RB) \leq 1$ since E is a binary random variable, $\mathsf{H}(A|B,R,E=1) = 0$ since $E = 1$ imply R and B determine A and $\Pr[E=0] \leq \delta$. □

Proposition 7 (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 8. *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. □

- Sample two random $B-$ary trees T_1, T_2 rooted at vertices 1 and 2 with $n/2$ leaves.
- Let the graph be empty to begin with.
- For $i \geq 0$, the i^{th} batch of updates correspond to adding edges in E_i in some arbitrary order to the current graph. Let U_i be the random variable denoting the i^{th} batch of updates.
- Let Q denote the random variable for the query that asks for connectivity between a random pair of leaves.

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

3 Dynamic Connectivity

We first show that the space of the data structure can be assumed to be polynomial in the number of operations, query time and the update time, with incurring a small error to the data structure.

Proposition 9. *Let N be the total number of operations to the data structure. Then, the data structure can be simulated with at most $N^3(t_u + t_q)^2$ number of cells with additional error of $\frac{1}{N}$.*

Proof. Consider a random hash $g : [s] \to \left[N^3(t_u + t_q)^2\right]$, where s is the total number of cells used by the data structure. Every access i to the data structure is simulated with access to $g(i)$. The probability that two distinct cells hash to the same number is at most $\frac{1}{N}$. This adds an additional error of $\frac{1}{N}$ to the data structure. □

Remark: Without loss of generality, we can take $t_q \leq n^2$ where n is the number of vertices in the graph. This is because, one can maintain the indicator vector for the edge set using $\binom{n}{2}$ cells and can answer connectivity queries in $\binom{n}{2}$ time and updates just take one write.

Before we show the proof of Theorem 1, we establish a fact about entropy of a subset of cells written by the data structure.

Proposition 10. *Let C be a subset of the cells, s be the number of cells. Then, $\mathsf{H}(C) \leq |C|(\log s + w)$.*

Proof. To encode the set C, we have to write down its contents along with the memory locations. It takes up at most $\log \binom{s}{|C|} \leq |C| \log s$ bit to encode the locations of C. Since each cell is w bits long, it takes up $w|C|$ bits to encode its contents. □

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

3.1 Proof of Theorem 1

We first present a hard distribution on a sequence of updates and queries. Set $B = (t_u \log n)^2$, $d = \log_B \frac{n}{2}$. The underlying graph will be a forest consisting of 2 B-ary trees T_1, T_2 rooted at vertices 1 and 2 with $n/2$ leaves each (Figure 2).

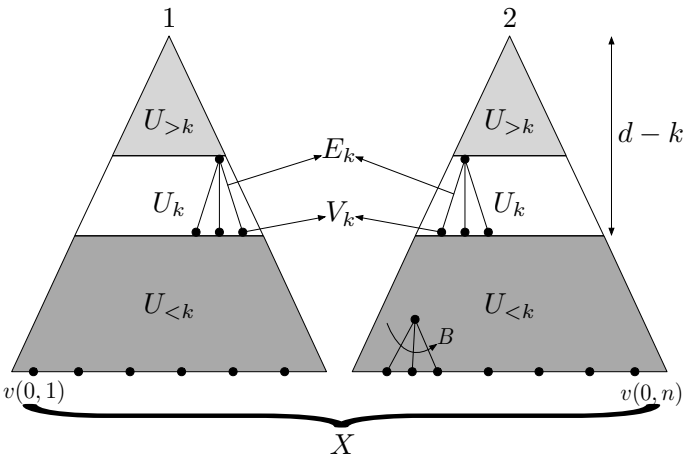

Figure 2: The input graph

The depth of a vertex $d(v)$ is the distance from its root. For every $k \in \{0, 1, \cdots, d\}$, let $V_k = \{v(k,1), \cdots, v(k, \frac{n}{B^k})\}$ denote the set of vertices at depth $d-k$. For $k \in \{0, \cdots, d-1\}$, let $E_k = \{(u,v) | u \in V_k, v \in V_{k+1}, (u,v) \in E(T_1) \cup E(T_2)\}$, where $E(T_1), E(T_2)$ are set of edges in trees T_1, T_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 a data structure on this input sequence.

Let $X = (X_1, \cdots, X_n)$ where $X_i = \mathsf{Conn}(v(i,0), 1)$. By Proposition 9, the data structure is $\frac{1}{16}$-error and has at most $8n^3(t_u + t_q)^2$ cells.

Lemma 11. $\mathsf{H}(X|U_{-k}) \geq \frac{n}{B^k} - 1 - \log\sqrt{\frac{n}{B^k}}$.

Proof. The input distribution dictates that $X|U_{-k}$ is a balanced n/B^k bit vector, drawn uniform 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 7. □

Define C_i to be the random variable denoting the set of cells written during batch i, and not overwritten by future updates $U_{>i}$.

Let $p_i = \Pr_{UQ}[Q \text{ probes a cell in } C_i]$. We have, $t_q = \sum_{i=0}^{d-1} p_i$. The following lemma imply the lower bound on t_q as stated in Theorem 1.

Lemma 12. *$p_k \geq \frac{1}{16} - o(1)$ for all $k \in \{0, \cdots d-2\}$.*

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

Lemma 13. *For every $k \in \{0, 1, \cdots, d-2\}$, $\mathsf{H}(X|C_{>k}U_{-k}) \geq n/B^k(1 - o(1))$.*

Proof. By the chain rule of entropy,

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

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

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

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

Since t_u is the worst case update time, we can now upper bound $|C_{>k}|$ by $|U_{>k}|t_u$. Proposition 10 and the upper bound on $|C_{>k}|$ imply,

$$\mathsf{H}(C_{>k}) \leq n/B^{k+1}\frac{B}{B-1}t_u(3\log(2n(t_u+t_q)) + 2\log 2n) = o(n/B^k),$$

which follows from the choice of B and $t_q \leq 4n^2$. □

Lemma 14. *Let $k \in \{0, 1, \cdots, d-2\}$ and S be a uniform random subset of $[n]$ of size 2. Then,*

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

Proof. $X|U_{-k}$ denote the connectivity labels of an equivalence class of size n/B^k, where 2 leaves belong to the same class if their parents at depth $d-k$ are the same. The projection S corresponds to a projection onto the equivalence classes given by the set of parents of elements in S at depth $d-k$. The distribution on the projection $S' \subseteq [n/B^i]$ 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.

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

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

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

The bound on $\mathsf{H}(X|C_{>k}U_{-k})$ from Lemma 13 and the fact that $B^k/n = o(1)$ completes the proof. □

Lemma 15. *For every $k \in \{0, 1, \cdots, d-2\}$, $S \subset [n]$ with $|S| = 2$,*

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

Proof. Let

$$E = \begin{cases}1, & \text{if Q queries into } C_k \\ 0, & \text{if Q does not query into } C_k\end{cases}$$

Let $i, j \in S$. By the chain rule of entropy, we have

$$\begin{aligned}\mathsf{H}(X_S|C_{>k}U_{-k}) &\leq \mathsf{H}(E|C_{>k}U_{-k}) + \mathsf{H}(X_S|C_{>k}U_{-k}, E) \\ &\leq h(p_k) + 2p_k + \mathsf{H}(X_S|C_{>k}U_{-k}, E = 0),\end{aligned}$$

where the last equality follows from $\Pr[E = 1] = p_k$ and $\mathsf{H}(X_S|C_{>k}U_{-k}, E = 1) \leq 2$.

Let E' be the indicator variable for the error in the output of $\mathsf{Conn}(i, j) = X_i \oplus X_j$. $E, E' = 0$ imply the algorithm knows the output $\mathsf{Conn}(i, j)$ without querying into C_k. Since X_S is supported on a universe of size 2, $H(X_S|C_{>k}U_{-k}, E = 0, E' = 0) \leq 1$.

By the chain rule of entropy,

$$\begin{aligned}\mathsf{H}(X_S|C_{>k}U_{-k}, E = 0) &\leq \mathsf{H}(E'|C_{>k}U_{-k}, E = 0) + \mathsf{H}(X_S|C_{>k}U_{-k}, E = 0, E') \\ &\leq h\left(\frac{1}{16}\right) + 1 + \frac{1}{16}H(X_S|C_{>k}U_{-k}, E = 0, E' = 1) \\ &\leq h\left(\frac{1}{16}\right) + \frac{1}{8},\end{aligned}$$

since, $\Pr[E' = 1] \leq 1/16$. □

Proof of Lemma 12. Lemma 14 implies that

$$\forall k \in \{0, \cdots, d-2\}, \mathop{\mathbb{E}}_{i,j\in[n]}[\mathsf{H}(X_iX_j|C_{>k}U_{-k})] \geq 2 - o(1).$$

Putting together the fact that $h\left(\frac{1}{16}\right) + \frac{1}{16} < 0.5$ and Lemma 15, we conclude the $h(p_k) + 2p_k \geq 1/2 - o(1)$ for every $k \in \{0, \cdots, d-2\}$. It follows that $p_k > 1/16 - o(1)$ which completes the proof of Lemma 12. □

3.2 Threshold Phenomenon in Dynamic Connectivity

We present the proof of Theorem 2.

Proof of Theorem 2. We use the hard distribution defined in [PT11] to prove the lower bound.

Set $B = (t_u \log n)^2$, $h = \log_B n$. Generate a graph as in Figure 3. Sample $n^{1-\delta}$ leaves $l_1, \cdots, l_{n^{1-\delta}}$ uniform at random. Let $R(l_i) \in \{v_1, \cdots, v_{n^\delta}\}$ be the random variable denoting the root of the leaf l_i. Let $\mathcal{L} = \{l^1, \cdots, l^{n^{2-\delta}}\}$ be the set of all leaves and $L = \left(R(l^1), \cdots, R(l^{n^{2-\delta}})\right)$ be a random variable denoting the label of roots of all leaves.

Let C^* be the set of cells read/written during the executing of the following operations. $\forall i \in [n^{1-\delta}]$, perform $\mathsf{Add}(l_i, R(l_i))$. Then, $\forall i \in [n^\delta - 1]$ execute $\mathsf{Conn}(v_i, v_{i+1})$ and if the output is 0, perform $\mathsf{Add}(v_i, v_{i+1})$. By Proposition 9, the data structure is $\frac{1}{3n^\delta}$-error and has at most $8n^{6-3\delta}(t_u + t_q)^2$ cells. By the union bound, the probability that the data structure is correct in answering all $\mathsf{Conn}(,)$ queries is at least 2/3.

Observation 16. $\mathbb{E}[|C^*|] \leq 2n^{1-\delta}t_u + n^\delta t_q$.

Proof. Note that the total number of $\mathsf{Add}(,)$ performed at most $2n^{1-\delta}$. The total number of $\mathsf{Conn}(,)$ executed is n^δ. □

Lemma 17. $\mathbb{E}[|C^*|] = \Omega\left(\delta n^{1-\delta} \log_B n\right)$.

- Sample $n^{1-\delta}$ complete $B-ary$ trees of height h.

Let $r_1, \cdots, r_{n^{1-\delta}}$ be the roots of the trees.

- Let $v_1, \cdots, v_{n^\delta}$ be n^δ vertices(disjoint from the vertices of the trees sampled so far) and edges incident to them are $\{(v_i, r_j) | (i-1)n^{1-2\delta} \leq j < in^{1-2\delta}\}$ for $i \in [n^\delta]$.

Let E_i be the set of edges between vertices at depth $h-i+1, h-i+2$. The sequence of updates are as follows,

- The i^{th} batch of updates corresponding to adding edges E_i in some arbitrary order. $i \in [h]$.
- U_{h+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, h, n, δ

Proof. Let $C_1, \cdots, C_{h+1}$ be such that C_i is the set of cells written by an $\mathsf{Add}(,) \in U_i$ and not overwritten by any updates in $U_{>i}$. We fix U_{-i}. Take $i \leq (1-\delta)\log_B n$.

Claim 18. $\mathsf{H}(L|U_{-i}) \geq \frac{\delta n^{2-\delta}\log n}{B^i}(1-o(1))$.

Proof. The total number of ways of partitioning a set of size $t = n^{2-\delta}/B^i$ 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

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

where the third equality followed from Proposition 7. Now $n^\delta \cdot O\left(\log\frac{t}{n^\delta}\right) = o(t\log n^\delta)$ by the choice of upper bound on i. □

Lemma 19. $\mathsf{H}(L|U_{-i}, C_{>i}) \geq \frac{\delta n^{2-\delta}\log n}{B^i}(1-o(1))$.

Proof. We have,

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

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

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

Since t_u is the worst case update time, we can now upper bound $|C_{>i}|$ by $|U_{>i}|t_u$. Proposition 10 and the upper bound on $|C_{>i}|$ imply,

$$\mathsf{H}(C_{>k}) \leq n^{2-\delta}/B^{i+1}\frac{B}{B-1}t_u(3\log(2n^{2-\delta}(t_u+t_q)) + 2\log 2n^{2-\delta}) = o(n^{2-\delta}/B^i),$$

which follows from the choice of B and $t_q \leq n^4$. The above bound and Claim 18 imply the lemma. □

Lemma 20. *For S drawn uniformly from $\{s|s \subseteq [n^{2-\delta}], |s| = n^{1-\delta}\}$,*

$$\mathsf{H}(L_S|S, U_{-i}, C_{>i}) \geq \delta n^{1-\delta}\log n\,(1-o(1))\,.$$

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

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

where the last inequality follows from Proposition 8. Therefore $\Pr[i \in S'] \geq \frac{B^i}{n} - \frac{B^{2i}}{n^2}$. Applying Shearer's lemma gives, $H(L^i_{S'}|S', U_{-i}, C_{>i}) \geq \left(\frac{B^i}{n} - \frac{B^{2i}}{n^2}\right)\cdot\frac{\delta}{B^i}\left(n^{2-\delta}\log n - o(n^{2-\delta}\log n)\right)$, as desired. We used the fact that the upper bound on i implies $B^i/n = o(1)$. □

Let $C^*_i \subset C_i$ be the set of cells within C^*. By definition, $\mathbb{E}\left[|C^*|\right] = \sum_{i=1}^h \mathbb{E}\left[|C^*_i|\right]$.

Lemma 21. $\mathsf{H}(L_S|S, U_{-i}, C_{>i}) \leq \mathbb{E}\left[|C^*_i|\right]O(\log n) + 1 + \frac{1}{3}\delta n^{1-\delta}\log n$.

Proof. Fix $S, U_{-i}, C_{>i}$. Consider the following decoding procedure for L_S. Let S be the randomness used by the data structure. For every $a_1, \cdots, a_{n^{1-\delta}} \in [n^\delta]$ and $j \in [n^{1-\delta}]$, we execute $\mathsf{Add}(l_j, a_j)$. Then, $\forall k \in [n^\delta - 1]$ execute $\mathsf{Conn}(v_k, v_{k+1})$ and if the output is 0, perform $\mathsf{Add}(v_i, v_{i+1})$. If the algorithm were correct with all query answers, it would output 0 for every query if and only if $a_i = R(l_i)$. By the definition of C^*_i, we know that every read/write into any cell in C_i is into C^*_i when $a_1 = R(l_1), \cdots, a_{n^{1-\delta}} = R(l_{n^{1-\delta}})$. Having access to S helps the algorithm execute every update/query. Knowing $U_{<i}$, the algorithm can generate $C_{<i}$ and hence any reads into $C_{<i}$ is made possible. By the guarantee of the data structure, in expectation over S, we have that with probability at least 2/3, all query answers are correct. We know that when all queries are correct, we will be able to recover $R(l_1), \cdots, R(l_{n^{1-\delta}})$. By Proposition 6 and $\mathsf{H}(L_S|S) \leq \delta n^{1-\delta}\log n$, we conclude that $\mathsf{H}(L_S|S, U_{-i}, C_{>i}) \leq \mathsf{H}(C^*_i) + 1 + \frac{1}{3}\delta n^{1-\delta}\log n$. We now upper bound $\mathsf{H}(C^*_i)$. To encode C^*_i, we have to specify the memory locations and its contents. By Proposition 10, it is upper bounded by $\mathbb{E}\left[|C^*_i|\right]\cdot O(\log n + \log t_u)$ bits. □

Lemma 21 and Lemma 20 imply $\mathbb{E}[|C_i^*|] = \Omega\left(\delta n^{1-\delta}\right)$. Since it holds for every fixing of $i \in [d]$, we get the desired bound on $\mathbb{E}[|C^*|]$. This completes the proof of Lemma 17. □

Lemma 17 and Observation 16 give,

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

□

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.

[CSRL01] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms.* McGraw-Hill Higher Education, 2nd edition, 2001.

[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

http://eccc.hpi-web.de