Low-Contention Data Structures

[Extended Abstract]

James Aspnes*
Department of Computer Science
Yale University
New Haven, CT 06511
aspnes@cs.yale.edu

David Eisenstat†
eisenstatdavid@gmail.com

Yitong Yin‡
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
yinyt@nju.edu.cn

ABSTRACT

We consider the problem of minimizing contention in static dictionary data structures, where the contention on each cell is measured by the expected number of probes to that cell given an input that is chosen from a distribution that is not known to the query algorithm (but that may be known when the data structure is built). When all positive queries are equally probable, and similarly all negative queries are equally probable, we show that it is possible to construct a data structure using linear space s, a constant number of queries, and with contention $O(1/s)$ on each cell, corresponding to a nearly-flat load distribution. All of these quantities are asymptotically optimal. For *arbitrary* query distributions, the lack of knowledge of the query distribution by the query algorithm prevents perfect load leveling in this case: we present a lower bound, based on VC-dimension, that shows that for a wide range of data structure problems, achieving contention even within a polylogarithmic factor of optimal requires a cell-probe complexity of $\Omega(\log \log n)$.

Categories and Subject Descriptors

E.1 [**Data Sstructures**]; F.1.2 [**Computation by Abstract Devices**]: Modes of Computation—*Parallelism and concurrency*

General Terms

Algorithms, Performance, Theory

*Supported in part by NSF grant CNS-0435201.

†Part of the work was done while David Eisenstat was a graduate student at Brown University.

‡Supported by the National Science Foundation of China under Grant No. 60721002. Part of the work was done while Yitong Yin was a graduate student at Yale University.

SPAA'10, June 13–15, 2010, Thira, Santorini, Greece.

Keywords

Memory contention, data structure, cell-probe model

1. INTRODUCTION

For shared-memory multiprocessors, memory contention measures the extent to which processors might access the same memory location at the same time, and is one of the main issues for realistic systems [9,13] and theoretical models [2,7]. In [6], a theoretical model is introduced by Dwork *et al.* to formally address the contention costs of algorithmic problems. In this paper, we propose to study the contention cost of a data structure, which measures how many queries to the data structure might simultaneously access the same memory cell. To avoid the question of how many queries to the data structure are running at the same time, we measure contention indirectly, by counting the expected number of probes to a given cell for each individual query. The expected number of probes to the cell for some fixed number m of simultaneous queries can then be bounded using linearity of expectation.

With binary search, for example, the entry in the middle of the table is accessed on every query, as is the cell storing the hash function or root-level index with perfect hashing and similar index-based data structures. Depending on the query distribution, the remaining load may be balanced almost as badly over the other cells.

We consider how to avoid this problem in case of *static* data structures, where the data structure is built in advance by a construction algorithm that may know the query distribution, but queries are performed by a uniform algorithm that does not (although it may use randomization itself to spread the query load more evenly).

The assumption that the query algorithm does not know the query distribution is natural. Often the query distribution will be highly correlated with the contents of the data structure, as in our simplest case where we consider a uniform distribution on successful queries to a static dictionary. Providing the query distribution to the query algorithm in such a case would, in effect, give it significant information about the contents of the data structure.

1.1 Model

Formally, a data structure problem is a function $f : Q \times \mathcal{D} \to \{0,1\}$, such that for every query $x \in Q$ and every data set $S \in \mathcal{D}$, $f(x,S)$ specifies the answer to the query x to data set S. A classic problem is the **membership**

problem, where $Q = [N]$ and $\mathcal{D} = \binom{[N]}{n}$ for some $N \gg n$, and $f(x, S) = 1$ if and only if $x \in S$.

We assume that the query $x \in Q$ follows some probability distribution q over Q.

For any data set $S \in \mathcal{D}$ and any query distribution q over Q, a table $T_{S,q} : [s] \to \{0,1\}^b$ of s cells, each of which contains b bits, is prepared. Given a query x, a probabilistic cell-probing algorithm $\mathcal{A}$ computes the value of $f(x, S)$ by making t randomized adaptive cell-probes $I_x^{(1)}, I_x^{(2)}, \ldots, I_x^{(t)} \in [s]$. The algorithm $\mathcal{A}$ may depend on f, but not on S or q (except to the extent that later probes may depend on the outcome of earlier probes, whose results might encode information about S and q).

The **contention** of a cell is the expected number of probes to the cell during one execution of $\mathcal{A}$. This will be equal to the probability that the cell is probed at all, provided $\mathcal{A}$ is sensible enough not to probe the same cell twice, but it is easier to work with expectations. In more detail:

DEFINITION 1. *For a fixed table $T_{S,q}$, for a query X chosen randomly from Q according to the distribution q, the sequence of cell-probes is $I_X^{(1)}, I_X^{(2)}, \ldots, I_X^{(t)}$. Let $Y^{(t)}(x, j)$ be the 0-1 valued random variable indicating whether $I_x^{(t)} = j$. The contention of cell j at step t is defined by*

$$\Phi_t(j) := \mathrm{E}\left[Y^{(t)}(X, j)\right],$$

where the expectation is taken over both X and the random $I_x^{(t)}$. The total contention of cell j is $\Phi(j) := \sum_t \Phi_t(j)$.

It is obvious that $\sum_j \Phi_t(j) = 1$, therefore $\frac{1}{s} \le \max_j \Phi_t(j) \le 1$. Ideally, we want $\max_j \Phi_t(j)$ to approach $\frac{1}{s}$.

A **balanced cell-probing scheme** is defined as follows:

DEFINITION 2. *An (s, b, t, ϕ)-balanced-cell-probing scheme for problem $f : Q \times \mathcal{D} \to \{0,1\}$ is a cell-probing scheme such that for any $S \in \mathcal{D}$ and any probability distribution q over Q, a table $T_{S,q} : [s] \to \{0,1\}^b$ is constructed, such that for any query $x \in Q$, the algorithm returns $f(x, S)$ by probing at most t cells, and for a query $x \in Q$ generated according to the distribution q, the contention $\Phi_k(j)$ is bounded by ϕ for any $1 \le k \le t$ and any $j \in [s]$.*

Such schemes have the very strong property that not only is contention bounded across an execution of the query algorithm, but each individual step gives low contention.

Given a fixed table $T_{S,q}$, we can summarize the contention succinctly using linear algebra. Let P_t be a $|Q| \times s$ matrix with $P_t(x, j) := \Pr[I_X^{(t)} = j] = \mathrm{E}[Y^{(t)}(X, j)]$. The contention on all cells can be computed by $\Phi_t = qP_t$, specifically,

$$\begin{aligned}\Phi_t(j) &= \mathrm{E}\left[Y^{(t)}(X, j)\right] \\ &= \sum_{x \in Q} \Pr[X = x] \cdot \mathrm{E}\left[Y^{(t)}(x, j)\right] \\ &= \sum_{x \in Q} q_x \cdot P_t(x, j).\end{aligned}$$

Finally, for our lower bound, it will be helpful to consider data structure problems from the perspective of communication complexity. In this view, a data structure is a communication protocol between an adaptive player Alice for the cell-probing algorithm and an oblivious player Bob for the table. The input to Bob is a pair (S, q), and the input to Alice is a query $x \in Q$, which is generated according to the distribution q. Together they compute $f(x, S)$ via communication. The contention then counts the probability of each type of message sent by Alice.

1.2 Our contributions

This paper makes following contributions:

- We formalize a natural and interesting problem: memory contention caused by concurrent data structure queries. We introduce contention to the classic cell-probe model. In our model, contention is measured by the chance that a memory cell is probed during the execution of the cell-probe algorithm. This level of abstraction allows us to study the trade-off between the contention and the complexity of data structures without regard to specific contention resolution schemes.

- We note an especially interesting class of query distributions: distributions that are uniform over both the set of positive queries and the set of negative queries (but not necessarily uniform over all queries). We introduce a linear-size, constant-time cell-probing scheme for the membership problem, with maximum contention $O(1/n)$. It is easy to see that this data structure is asymptotically optimal in all three parameters.

- We study data structures with arbitrary query distribution. For this general case, we prove a lower bound on any balanced cell-probing scheme satisfying a certain natural technical restriction. The lower bound is a time-contention trade-off: for any problem which has a non-degenerate subproblem of size n, if the contention is within a Polylog(n) factor of optimal, the time complexity is $\Omega(\log \log n)$. This directly implies the same lower bound for the membership problem.

1.3 Related work

Our first upper bound is based on the well-known FKS construction of Fredman *et al.* [8] and subsequent work by Dietzfelbinger and Meyer auf der Heyde extending these results to the dynamic case [3–5]; we will refer to this latter construction, as described in [4], by DM.

The FKS construction is a static data structure for the membership problem, based on a two-level tree of hash tables, with linear space and constant lookup time.

In DM, the hash functions used in FKS are replaced with a new family that gives a more even distribution of load across the second layer of the tree, which is used to get bounded worst-case update costs for the dynamic case. In [3] and [5], DM is implemented in the PRAM model and the model of a complete synchronized network of processors respectively. Both implementations optimize the contention on individual processors, but do not consider the contention on individual memory locations.

Membership can also be solved with optimal time and space complexity using cuckoo hashing [12]; as with FKS and DM, the contention of the standard implementation is high, mostly because all queries read the hash function parameters from the same locations.

For FKS, DM, and cuckoo hashing, contention can be decreased by storing the hash function redundantly. Under the assumption that the query is distributed uniformly

within both the positive set and the negative set, this gives a maximum contention of $\Theta(\sqrt{n})$ times optimal for FKS, and $\Theta(\ln n / \ln \ln n)$ times optimal for DM and cuckoo hashing; while for arbitrary query distributions, the contentions can be arbitrarily bad. This is not surprising, given that none of these data structures are designed with memory contention in mind; nonetheless, we show that it is possible to do substantially better.

2. LOW-CONTENTION UNIFORM MEMBERSHIP QUERIES

Let $N = |U|$ be the size of the universe. We assume that $N \geq n^2$, and each cell in the table contains a b-bit word, where $b = \log_2 N$.

THEOREM 3. *For the membership problem of n elements, with the assumption that the query is uniformly distributed within both positive queries and negative queries, there exists an $(O(n), b, O(1), O(1/n))$-balanced-cell-probing scheme.*

Given a data set $S \in \binom{U}{n}$, the data structure can be constructed in expected $O(n)$ time on a unit-cost RAM.

To see how our data structure works, it may help to start by considering the query procedure for FKS hashing. FKS hashing works by taking a standard hash table and using a secondary perfect hashing scheme within each of $O(n)$ "buckets" to resolve collisions between elements hashed to the same bucket. Even though the largest bucket may contain $O(\sqrt{n})$ elements, and the size of the i-th bucket is proportional to the *square* n_i^2 of the number of items n_i in that bucket, because most buckets are small, the sum of these squares is likely to be linear in n.

FKS guarantees that all queries finish in exactly three probes: the first probe reads the parameters of the hash function; the second reads a pointer to the "bucket" in which the target item will be found, as well as information about the size of the bucket and the perfect hash function used within the bucket; and the third reads the actual element. This produces contention 1 on the cell for the first probe and contention $\Theta(n_i/n)$ on the cell for the second probe; both are much worse than our goal of $O(1/n)$.

We can reduce the contention for the first probe by replication; instead of probing a single cell, we probe one of n identical copies. The second probe is trickier; we would like to replicate the information for large buckets, but the query algorithm does not know which buckets are large.

Our approach is to organize the buckets into $\Theta(n/\log n)$ groups of $\Theta(\log n)$ buckets each. While individual buckets may vary significantly in size, we can show that when using the hash functions of DM [4], the total size of each group will be $O(\log n)$ with reasonably high probability. A bit-vector encoding allows us to indicate the size of all buckets in a group in a single $O(\log n)$-bit cell, which is replicated $O(\log n)$ times to reduce contention to $O(1/n)$. Knowing the size of each bucket in the group, the query algorithm can deduce the storage range for the replicated headers of the target bucket, read the relevant header information (including both a pointer to the actual location of the bucket and the parameters of its secondary hash function) from a randomly-distributed probe, and finally use the bucket's perfect hash function to find the target element. This four-phase procedure requires a constant number of probes and still uses only $O(n)$ space with $O(1/n)$ contention, for either uniform positive or uniform negative queries.

2.1 Hash families

In [1], **universal hash classes** were introduced. For $d \geq 2$, a family of functions from U to $[m]$ is d-wise independent (or d-**universal**) if for any d distinct elements $x_1, x_2, \ldots, x_d$ from U, the hash values $h(x_1), h(x_2), \ldots, h(x_d)$ are uniformly and independently distributed over $[m]$.

Let $\mathcal{H}_m^d$ denote a d-wise independent hash family of hash functions from U to $[m]$. It is well known that if $d \geq 2$ and $m \geq n^2$, for any $S \in \binom{U}{n}$, with at least $\frac{1}{2}$ probability a uniformly random $h \in \mathcal{H}_m^d$ maps each element in S to a distinct value; i.e., it is a *perfect hash function*.

We use the following hash family, first introduced in [4].

DEFINITION 4 (DM [4]). *For $f \in \mathcal{H}_m^d$, $g \in \mathcal{H}_r^d$, and $z \in [m]^r$ the hash function $h_{f,g,z} : U \to [m]$ is defined by*

$$h_{f,g,z}(x) := (f(x) + z_{g(x)}) \bmod m.$$

The hash family $\mathcal{R}_{r,m}^d$ is

$$\mathcal{R}_{r,m}^d := \{h_{f,g,z} \mid f \in \mathcal{H}_m^d, g \in \mathcal{H}_r^d, z \in [m]^r\}.$$

Given a hash function and a set of elements, we define the buckets and loads as follows.

DEFINITION 5. *For $h : U \to [m]$, $S \subseteq U$, and $i \in [m]$, the i-th bucket $B(S,h,i) := \{x \in S \mid h(x) = i\}$, and the load of the i-th bucket is $\ell(S,h,i) := |B(S,h,i)|$.*

The following theorem is from [11]. It bounds the deviation of the sum of a 0-1 valued d-wise independent sequence.

THEOREM 6 (COROLLARY 4.20, [11]). *Let $X_1, \ldots, X_n$ be 0-1 valued, d-wise independent, equidistributed random variables. Let $X = \sum_i^n X_i$. If $d \leq 2E[X]$, then*

$$\Pr[X - E[X] > t] \leq O\left(\frac{(E[X])^{d/2}}{t^d}\right).$$

The following is a special case of the Hoeffding's theorem [10].

THEOREM 7 (HOEFFDING). *Let $Y_1, \ldots, Y_r$ be independent random variables with range of values in $[0,d]$. Let $Y = \sum_i^r Y_i$, and $c > e$ be some constant. If $cE[Y] \leq rd$, then*

$$\Pr[Y \geq cE[Y]] \leq \left(\frac{e}{c}\right)^{\frac{c}{d}E[Y]}.$$

For d-universal hash families, the following theorem holds.

THEOREM 8 (FACT 2.2, [4]). *Let S be a fixed set of n elements. Let f be chosen from $\mathcal{H}_m^d$ uniformly at random, where $d > 2$ is a constant and $m \leq 2n/d$. Then*

$$\Pr[\forall i \in [m], \ell(S,f,i) \leq d] \geq 1 - n \cdot (2n/m)^d.$$

The following lemma characterizes the load distribution of functions from various families.

LEMMA 9 (EXTENDED FROM [4, 8, 11]). *Fix an $S \in \binom{U}{n}$. Let $c > e$ and $d > 2$ be constants. The following holds:*

1. *For $r = n^{1-\delta}$ where $\frac{2}{d+2} < \delta < 1 - \frac{1}{d}$, and g from $\mathcal{H}_r^d$, $\Pr[\forall i \in [r], \ell(S,g,i) \leq cn/r] \geq 1 - o(1)$.*
2. *For $m = \frac{n}{\alpha \ln n}$ where $\alpha > \frac{d}{c(\ln c - 1)}$, and h' from $\mathcal{R}_{r,m}^d$, $\Pr[\forall i \in [m], \ell(S,h',i) \leq cn/m] \geq 1 - o(1)$*

3. (FKS condition) For $s = \beta n$ where $\beta \geq 2$, and h from $\mathcal{R}^d_{r,s}$, $\Pr\left[\sum_{i\in[s]}(\ell(S,h,i))^2 \leq s\right] \geq \frac{1}{2}$.

PROOF. Let $S = \{x_1, x_2, \ldots, x_n\}$.

1. For a fixed $j \in [r]$, let X_i be a 0-1 valued random variable that indicates whether $g(x_i) = j$, and let $X = \sum_i^n X_i$. It is obvious that $E[X] = \frac{n}{r} = n^\delta$. Due to Theorem 6,

$$\begin{aligned}\Pr\left[X - n^\delta \geq (c-1)n^\delta\right] &\leq O\left(n^{\delta d/2}/n^{\delta d}\right)\\ &= O\left(n^{-\delta d/2}\right).\end{aligned}$$

Therefore,

$$\begin{aligned}&\Pr\left[\exists j \in [r], \ell(S,g,j) > \frac{cn}{r}\right]\\ \leq\ & r \cdot \Pr\left[\ell(S,g,j) > cn^\delta\right]\\ \leq\ & n^{1-\delta} \cdot \Pr\left[X - n^\delta > (c-1)n^\delta\right]\\ \leq\ & O\left(n^{1-\delta-\delta d/2}\right) = o(1).\end{aligned}$$

2. We assume that h' is defined by (f,g,z) where f and g are randomly drawn from $\mathcal{H}^d_m$ and $\mathcal{H}^d_r$ respectively, and z is chosen uniformly from $[m]^r$. We define the following two events

$$\begin{aligned}&\mathcal{E}_1 : \forall i \in [r], \ell(S,g,i) \leq cn/r;\\ &\mathcal{E}_2 : \forall i \in [r], \forall j \in [m], \ell(B(S,g,i),f,j) \leq d.\end{aligned}$$

Due to the first part, $\mathcal{E}_1$ holds with probability $1 - o(1)$. Conditioning on $\mathcal{E}_1$, according to Theorem 8, with probability $1 - O(n^{\delta(d+1)}/m^d)$, it holds that

$$\forall j \in [m], \ell(B(S,g,i),f,j) \leq d.$$

By union bound, event $\mathcal{E}_2$ holds with probability

$$\begin{aligned}1 - O(n^{1-\delta} \cdot n^{\delta(d+1)}/m^d) &= 1 - O(n^{1-d(1-\delta)}(\ln n)^d)\\ &= 1 - o(1).\end{aligned}$$

Therefore,

$$\begin{aligned}\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] &= \Pr[\mathcal{E}_1] \cdot \Pr[\mathcal{E}_2 \mid \mathcal{E}_1]\\ &= (1 - o(1))(1 - o(1))\\ &= 1 - o(1).\end{aligned}$$

Conditioning on $\mathcal{E}_1 \wedge \mathcal{E}_2$, for any fixed $j \in [m]$, for $i = 1, 2, \ldots, r$, define random variable Y_i as

$$\begin{aligned}Y_i := |\{x \in S \mid\ & g(x) = i \text{ and}\\ & f(x) + z_{g(x)} \equiv j \pmod m\}|.\end{aligned}$$

Let $Y = \sum_i^r Y_i$. Note that

$$\begin{aligned}Y_i &= \ell(B(S,g,i),h',j)\\ &= \ell(B(S,g,i),f,(j - z_i + m) \bmod m),\end{aligned}$$

and $Y = \ell(S,h',j)$.

Because $\mathcal{E}_2$ holds, $Y_i \leq d$ for all $i \in [r]$, and Y_i are independent because z_i are independent.

$$\begin{aligned}&E[Y_i \mid f,g]\\ =\ & \sum_{z_i \in [m]} \frac{1}{m}\ell(B(S,g,i),f,(j - z_i + m) \bmod m)\\ =\ & \frac{1}{m}\sum_{k\in[m]} \ell(B(S,g,i),f,k)\\ =\ & \frac{1}{m}\ell(S,g,i).\end{aligned}$$

Therefore

$$\begin{aligned}E[Y] &= E\left[E\left[Y \mid f,g\right]\right]\\ &= \frac{1}{m}E\left[\sum_{i\in[r]}\ell(S,g,i)\right]\\ &= \frac{|S|}{m} = \frac{n}{m}.\end{aligned}$$

According to Theorem 7, it holds that

$$\Pr[Y \geq cn/m] \leq (e/c)^{c\alpha \ln n/d} = o(n^{-1}).$$

By union bound,

$$\begin{aligned}&\Pr[\forall j \in [m], \ell(S,h',j) \leq cn/m]\\ &= 1 - m \cdot \Pr[Y \leq cn/m]\\ &= 1 - o(1).\end{aligned}$$

Recall that the above holds when conditioning on $\mathcal{E}_1 \wedge \mathcal{E}_2$. Since $\Pr[\mathcal{E}_1 \wedge \mathcal{E}_2] = 1 - o(1)$, the event

$$\forall j \in [m], \ell(S,h',j) \leq cn/m$$

holds unconditionally with probability at least $(1 - o(1))(1 - o(1)) = 1 - o(1)$.

3. For every $i, j \in [n]$ where $i \neq j$, let X_{ij} be 0-1 valued random variable that indicates whether $h(x_i) = h(x_j)$. Let $X = \sum_{i\neq j} X_{ij}$ be the total number of ordered collision pairs. It is easy to see that

$$X = 2\sum_{i\in[s]}\binom{\ell(S,h,i)}{2} = \sum_{i\in[s]}(\ell(S,h,i))^2 - n.$$

Note that h is at least 2-wise independent, thus for any $i \neq j$, $E[X_{ij}] = \Pr[h(x_i) = h(x_j)] = 1/s$, thus $E[X] = n(n-1)/s$. Due to Markov's inequality,

$$\begin{aligned}\Pr\left[\sum_{i\in[s]}(\ell(S,h,i))^2 > s\right] = \Pr[X > s - n] &\leq \frac{E[X]}{(s-n)}\\ &\leq \frac{1}{\beta(\beta-1)}\\ &\leq 1/2.\end{aligned}$$

□

2.2 Data structure construction

Let $c = 2e$. For $d > 2$, choose appropriate constants α and β as stated in Lemma 9, and let $r = n^{1-\delta}$ and $m = \frac{n}{\alpha \ln n}$. In addition, choose an appropriate constant $\beta \geq 2$ to make $s = \beta n$ divisible by m.

Given any data set $S \in \binom{U}{n}$, uniformly choose $f \in \mathcal{H}_s^d$, $g \in \mathcal{H}_r^d$, and $z \in [s]^r$, and construct a uniformly random $h \in \mathcal{R}_{r,s}^d$ by letting $h(x) := (f(x) + z_{g(x)}) \bmod s$. Define a new hash function $h' : U \to [m]$ by $h'(x) = h(x) \bmod m$. Note that h' is a uniformly random function from the family $\mathcal{R}_{r,m}^d$ because m divides s. Specifically,

$$\begin{aligned} h'(x) &= \big(f(x) + z_{g(x)}\big) \bmod s \bmod m \\ &= \big(f(x) \bmod m + z_{g(x)} \bmod m\big) \bmod m. \end{aligned}$$

For uniform $f \in \mathcal{H}_s^d$ and uniform $z \in [s]^r$, $(f \bmod m)$ and $(z \bmod m)$ are uniform over $\mathcal{H}_m^d$ and $[m]^r$ respectively. Therefore, h' is uniform over $\mathcal{R}_{r,m}^d$.

We would like our hash function to satisfy the property:

$$\begin{aligned} \mathcal{P}(S) \quad := \quad \Big\{ \big(g, h', h\big) \in \mathcal{H}_r^d \times \mathcal{R}_{r,m}^d \times \mathcal{R}_{r,s}^d \;\Big|\; & \\ & \forall i \in [r], \ell(S, g, i) \le cn/r, \\ & \text{and } \forall i \in [m], \ell(S, h', i) \le cn/m, \\ & \text{and } \sum_{i \in [s]} \ell^2(S, h, i) \le s \Big\} \end{aligned}$$

Due to Lemma 9, and by applying the union bound to all unwanted events, for the above g, h' and h, it holds that $(g, h', h) \in \mathcal{P}(S)$ with probability at least $1/2 - o(1)$. Therefore by repeatedly generating (g, h', h), we satisfy $\mathcal{P}(S)$ within expected $O(1)$ trials. Note that $\mathcal{P}(S)$ can be verified in $O(n)$ time in a unit-cost machine, thus a good hash function can be found within expected $O(n)$ time.

The data structure is organized in rows of cells where each row contains s cells. Let $T(i, j)$ represent the j-th cell in the i-th row in the data structure. T is constructed as follows:

- Let $a_0, a_1, \ldots, a_{2d-1}$ denote the $2d$ words that represent the two d-universal functions f and g. Let $T(i, j) = a_i$ for every $i \in [2d]$ and every $j \in [s]$. Let $T(2d, j) = z[j \bmod r]$ for every $j \in [s]$.

- We say that h assigns the n elements in S into s buckets, and h' arranges the buckets into m groups according to the congruence classes of h modulo m. For group $i \in [m]$, we define the *group-base-address* $GBA_S(i)$ as $GBA_S(0) = 0$ and

  $$GBA_S(i) = GBA_S(i-1) + \sum_{k \in [s/m]} \ell^2(S, h, km + i - 1).$$

  The vector GBA_S can be computed in $O(n)$ time in a unit-cost machine. Due to the property $\mathcal{P}(S)$, $GBA_S(i) \le s$ for any $i \in [m]$. Let $T(2d+1, j) = GBA_S(j \bmod m)$, i.e. each bucket stores the group-base-address of the group that the bucket belongs to.

- Let a *group-histogram* be a binary string where the load of each bucket in the group is represented consecutively in unary code separated by zeros.

  Each group contains $s/m = \alpha\beta \ln n$ buckets, and due to property $\mathcal{P}(S)$, each group contains at most $cn/m = c\alpha \ln n$ elements from S. Therefore the group-histogram uses at most $\alpha(\beta + c) \ln n$ bits. Let $\rho := \lceil \frac{\alpha(\beta+c)\ln n}{b} \rceil$. Observe that because $b = \Theta(\log n)$, $\rho = O(1)$. Let $a'_{0j}, a'_{1j} \ldots, a'_{\rho-1,j}$ denote the ρ words that store the group-histogram of group j.

  Let $T(2d+2+i, j) = a'_{i,(j \bmod m)}$, for $i = 0, 1, \ldots, \rho-1$, and for all $j \in [s]$.

- The last two rows are used to perfectly hash each bucket. Each bucket $i \in [s]$ owns $\ell^2(S, h, i)$ cells in each row. Due to $\mathcal{P}(S)$, the total space is at most s. The spaces owned by the buckets are organized in groups. If bucket i is the k-th bucket in group j, then the spaces owned by the buckets are sorted lexicographically. This can be done in a total $O(n)$ time in a unit-cost machine.

  In the $(2d + \rho + 1)$th row, for each individual bucket i, the perfect hash function h_i^* is stored repeatedly in the space owned by the bucket. In the $(2d + \rho + 2)$th row, the actual data in each bucket i is stored according to the hash function h_i^*.

The table T has $(2d + \rho + 2) = O(1)$ rows, each of which contains $s = O(n)$ words, for a total of $O(n)$ words. Each step of the construction costs $O(n)$ time, for a total of $O(n)$ time.

2.3 Queries and contention

We query whether x is in S with the following algorithm. Each random choice is assumed to be independent and uniform within its range.

1. For each $i \in [2d]$, choose $j \in [s]$, and read $T(i, j)$; this gives f and g. Next choose $k \in [s/r]$ and read $T(2d, kr + g(x))$, which stores $z_{g(x)}$. We can now compute $h = (f + z_g) \bmod s$ and $h' = h \bmod m$.

2. Choose k in $[s/m]$, and read $T(2d, km + h'(x))$, which stores $GBA_S(h'(x))$. For each $i \in [\rho]$ where $\rho = \lceil \frac{\alpha(\beta+c)\ln n}{b} \rceil$, choose some $j \in [s/m]$, and read $T(2d + 1 + i, jm + h'(x))$; we thus obtain the group-histogram group $h'(x)$. With the group-base-address and the group-histogram, the exact range of the address owned by bucket $h(x)$ can be determined: it runs from $i_{h(x)}$ to $i'_{h(x)}$ inclusively, where

   $$\begin{aligned} i_{h(x)} \quad &:= \quad GBA_S(h'(x)) \\ &\qquad + \sum_{k=0}^{\lceil h(x)/m \rceil - 1} \ell^2\left(S, h, km + h'(x)\right), \\ i'_{h(x)} \quad &:= \quad i_{h(x)} + \ell^2\left(S, h, h(x)\right) - 1. \end{aligned}$$

   All the values $\ell^2(S, h, km + h'(x))$ for $k \in [s/m]$ are stored in the group-histogram of group $h'(x)$.

3. If $i'_{h(x)} < i_{h(x)}$, the bucket $h(x)$ is empty: return 0. Otherwise, choose $j \in [i_{h(x)}, i'_{h(x)}]$ and read $T(2d + \rho + 1, j)$ to get the perfect hash function h^*. If $T(2d + \rho + 2, i_{h(x)} + h^*(x)) = x$, return 1, else return 0.

The correctness of the algorithm is guaranteed by the existence of hash functions with property $\mathcal{P}(S)$ and the existence of the perfect hashing scheme for each bucket, which is guaranteed. The query algorithm makes at most one probe to each row of T, thus the cell-probe complexity is $O(1)$.

For contention, we first consider the contribution of the positive queries. All events below are conditioned on the target element being in S. At each step before the last probe, an expected 1, $\frac{1}{n}\ell(S, g, i_1)$, $\frac{1}{n}\ell(S, h', i_2)$, or $\frac{1}{n}\ell(S, h, i_3)$ probes

are balanced over a range of size s, s/r, s/m, or $\ell^2(S, h, i_3)$ respectively, therefore due to property $\mathcal{P}(S)$, the maximum contention is $O(1/n)$. For the last probe, the perfect hash function sends each query to a distinct cell so the contention is obviously $O(1/n)$. Therefore, the total contention contributed by positive queries is at most $O(1/n)$.

LEMMA 10. *Let $\bar{S}$ denote $U \setminus S$, and $N = |U| = \omega(n)$. For any hash function $h : U \to [k]$ which is uniform over the domain, for sufficiently large n, $\forall i \in [k], \ell(\bar{S}, h, i) \leq 2(N-n)/k$.*

PROOF. Because h is uniform over the domain, $\ell(U, h, i) = N/k$ for any $i \in [k]$. For $N = \omega(n)$, it holds that $\ell(\bar{S}, h, i) = \ell(U, h, i) - \ell(S, h, i) \leq N/k \leq 2(N-n)/k$. □

Note that g, h', and h are all uniform over the domain. This is because any d-universal function must be 1-universal. Due to Lemma 10, the loads of negative queries to all types of buckets are asymptotically even. The same argument as above can be applied to bound the contention caused by negative queries to $O(1/n)$.

3. A LOWER BOUND FOR ARBITRARY QUERY DISTRIBUTIONS

In this section, we prove a cell-probe lower bound for low-contention data structures with arbitrary query distribution. The lower bound is on the cost of queries by an algorithm that does not know the distribution; however, the algorithm that constructs the data structure may known the distribution, and may optimize the data structure to minimize contention by encoding the information of query distribution in the data structure to guide the query algorithm.

The lower bound itself is proved based on the following intuition: the more uniform a random probe is, the less specific information it retrieves; but non-uniform probes will only result in low contention if the query algorithm already has some knowledge about the query distribution. So to obtain information about a specific query, the query algorithm must steadily increase its knowledge of the query distribution through increasingly non-uniform probes. By tracking how much information the query algorithm has about the query distribution (and thus how evenly spread out it must keep its probes), we can bound the query time subject to bounds on the contention.

We formalize this argument using VC-dimension [14], a measure of complexity of classification problems, by treating a data structure problem $f : Q \times \mathcal{D} \to \{0,1\}$ as a class of $|\mathcal{D}|$ many classifications of Q.

DEFINITION 11. *The **VC-dimension** of a data structure problem $f : Q \times \mathcal{D} \to \{0,1\}$, denoted by VC-dim$(f)$, is the maximum n such that there exists a set $\{x_1, x_2, \ldots, x_n\} \in \binom{Q}{n}$ such that for any assignment $y \in \{0,1\}^n$, there exists some $S \in \mathcal{D}$, with $f(x_i, S) = y_i$ for all i.*

It is easy to see that the VC-dimension of the membership problem is n, where n is the cardinality of the data set. This allows us to translate our results for problems of arbitrary VC-dimension into specific results for membership.

We consider a special class of cell-probing schemes $(T, \mathcal{A})$ whose cell-probing algorithm $\mathcal{A}$ satisfies a natural restriction described as follows.

DEFINITION 12. *A **table structure** is a mapping T which for any data set $S \in \mathcal{D}$ and any query distribution q over Q, specifies a table $T_{S,q} : [s] \to \{0,1\}^b$ of s cells, each of which contains b bits.*

*Given any query $x \in Q$, a **cell-probing algorithm** $\mathcal{A}$ returns $f(x, S)$ by making at most t^* randomized adaptive probes $I_x^{(1)}, I_x^{(2)}, \ldots, I_x^{(t^*)} \in [s]$ to the table $T_{S,q}$, such that the maximum contention satisfies $\Phi_t \leq \phi^*$ for any $t \leq t^*$, and for any fixed query x and any fixed table $T_{S,q}$, the random variables $I_x^{(t)}$ for all $t \leq t^*$ are jointly independent.*

The independence of cell-probes of $\mathcal{A}$ does not make $\mathcal{A}$ non-adaptive, because the independence holds only when the query and the table are both fixed. Note that in this sense all deterministic cell-probing algorithms (both adaptive or non-adaptive) satisfy this property, because once the table and the query are both fixed, the sequence of the cell-probes of a deterministic cell-probing algorithm are fixed, hence they are jointly independent. Informally speaking, for $\mathcal{A}$, the randomness is used only for balancing the cell-probes, but is not involved in the process of decision making. The upper bound presented in the previous section and any upper bounds constructed by the technique of distributing probes across multiple copies of critical cells are all included in this definition.

We prove the following lower bound theorem.

THEOREM 13. *For any data structure problem f with a VC-dimension VC-dim$(f) = n$, if there exists a cell-probing scheme $(T, \mathcal{A})$ as defined in Definition 12, with size of cell $b \leq \text{Polylog}(n)$ and contention $\phi^* \leq \frac{\text{Polylog}(n)}{s}$, then the cell-probe complexity $t^* = \Omega(\log \log n)$.*

The theorem is proved by first running n different instances of queries in parallel and then bounding the speed with which these parallel instances of the cell-probing algorithm gather information.

LEMMA 14. *If there exists a cell-probing scheme $(T, \mathcal{A})$ for the data structure problem f where $(T, \mathcal{A})$ and f are as in Theorem 13, then there exists a communication protocol between an algorithm $\mathcal{A}''$ and a black-box $\mathcal{B}$ which is specified as follows. The input to $\mathcal{B}$ is an arbitrary **stochastic vector** $q \in [0,1]^n$ that $\sum_{i=1}^n q_i \leq 1$, which is initially unknown to $\mathcal{A}''$. The communication between $\mathcal{A}''$ and $\mathcal{B}$ occurs in rounds.*

1. *At round t, $\mathcal{A}''$ specifies an $n \times s$ matrix P_t, called a **probe specification**, and sends it to $\mathcal{B}$, where P_t is adaptive to the information received previously by $\mathcal{A}''$, and for any $1 \leq i \leq n$, it holds that*

$$\sum_{j=1}^{s} P_t(i,j) \leq 1; \tag{1}$$

$$\max_{1 \leq j \leq s} P_t(i,j) \leq \frac{\phi^*}{q_i}. \tag{2}$$

2. *Upon receiving a P_t, $\mathcal{B}$ sends C_t bits to $\mathcal{A}''$, where C_t is a random variable satisfying*

$$\mathrm{E}[C_t] \leq b \cdot \sum_{j=1}^{s} \max_{1 \leq i \leq n} P_t(i,j), \tag{3}$$

where the expectation is conditioned on P_t, thus conditioned on all previous communication between $\mathcal{A}''$ and $\mathcal{B}$.

3. After t^* rounds, the expected number of bits received by $\mathcal{A}''$ is at least $n \cdot 2^{-2t^*}$ bits from $\mathcal{B}$.

PROOF. The idea of the proof is to run n instances of the cell-probing algorithm in parallel; together these instances form $\mathcal{A}''$. We observe that the cell-probes of each individual cell-probing algorithm can be specified by a probability distribution of probes over the table, and the joint distribution of the cell-probes of all n instances can be arbitrarily chosen by us as long as the marginal distribution of the cell-probe of each individual instance is the same as before, therefore (by our choice of the joint distribution) the total information obtained by $\mathcal{A}''$ in each round is bounded.

The details of the proof are given in Appendix A. □

The following two combinatorial lemmas are needed for the proof of Theorem 13.

LEMMA 15. *Let M be an $N \times n$ nonnegative matrix. Let $r = \sqrt{5\epsilon^{-1}\delta n \ln N}$. Assume that for every row $1 \le u \le N$, there exists a set $R_u \in \binom{\{1,2,\ldots,n\}}{r}$ of r entries such that $\sum_{i \in R_u} M(u,i) \le \delta$. Then there exists $q \in [0,1]^n$ that $\sum_i q_i = \epsilon$, such that for all $1 \le u \le N$, there exists $1 \le i \le n$, such that $M(u,i) < q_i$.*

PROOF. For each $1 \le u \le N$, sort $\{M(u,i) \mid i \in R_u\}$ by non-decreasing order and let $R'_u \subseteq \{1,2,\ldots,n\}$ be the indices of the smallest $\frac{r}{2}$ entries. It holds that $\forall i \in R'_u$, $M(u,i) \le \frac{2\delta}{r}$, as otherwise it contradicts the assumption that $\sum_{i \in R_u} M(u,i) \le \delta$.

It holds that for any choice of such $\{R'_u\}_{1 \le u \le N}$, there exists a $T \subseteq \{1,2,\ldots,n\}$, such that $|T| = \frac{2n \ln N}{r}$ and T intersects all R'_u. We prove this by the probabilistic method: let T be a uniformly random subset of $\{1,2,\ldots,n\}$ of size $\frac{2n \ln N}{r}$, thus each R'_u is missed by T with probability less than $(1 - r/2n)^{2n \ln N / r} < 1/N$, thus by the union bound, T intersects all R'_u with positive probability.

Fix such a T, define $q \in [0,1]^n$ as $q_i = \epsilon |T|^{-1} = \frac{r\epsilon}{2n \ln N}$ if $i \in T$, and $q_i = 0$ if otherwise. Therefore, $\sum_i q_i = \epsilon$, and for any $1 \le u \le N$, for such $i \in R'_u \cap T$, it holds that $M(u,i) \le \frac{2\delta}{r} < \frac{r\epsilon}{2n \ln N} = q_i$. □

LEMMA 16. *For any nonnegative $n \times s$ matrix P that $\sum_j P(i,j) \le 1$ for every i, let R be the largest subset of $\{1,2,\ldots,n\}$ that $\sum_{i \in R} \frac{1}{\max_j P(i,j)} \le s$. Then it holds that*

$$|R| \ge \sum_{j=1}^{s} \max_{1 \le i \le n} P(i,j).$$

PROOF. The sum $\sum_j \max_i P(i,j)$ chooses exactly s entries to sum up. Let A_i be the set of chosen columns in row i. Let $x_i := \sum_{j \in A_i} P(i,j)$. Note that $x_i \le \sum_j P(i,j) \le 1$. By the pigeonhole principle, for any $1 \le i \le n$,

$$|A_i| \ge \frac{\sum_{j \in A_i} P(i,j)}{\max_j P(i,j)} = \frac{x_i}{\max_j P(i,j)}.$$

Note that $\sum_i |A_i| = s$, thus $\sum_i \frac{x_i}{\max_j P(i,j)} \le \sum_i |A_i| = s$.

Therefore the sum $\sum_j \max_i P(i,j)$ can be written as

$$\sum_j \max_i P(i,j) = \sum_i \sum_{j \in A_i} P(i,j) = \sum_i x_i,$$

subject to the constraints that $\sum_i \frac{x_i}{\max_j P(i,j)} \le s$ and $x_i \le 1$. It is easy to see that the value of $\sum_i x_i$ is maximized when letting $x_i = 1$ for $i \in R$ and $x_i = 0$ for $i \notin R$, therefore $\sum_j \max_i P(i,j) = \sum_i x_i \le |R|$. □

Proof of Theorem 13:

Given the algorithm $\mathcal{A}''$ as described in Lemma 14, we will bound the speed that $\mathcal{A}''$ gathers information. Due to Lemma 14, $\mathcal{A}''$ is a decision tree in which the current node of depth $(t-1)$ has $N_t := 2^{C_{t-1}}$ children, each of which corresponds to a next probe specification P_t. We number these P_t by $u \in [N_t]$ and denote each as $P_t^{(u)}$, where u can be interpreted as the bit string received by $\mathcal{A}''$ at round $t-1$. We then inductively bound the next C_t.

Let $M^{(t)}$ be an $N_t \times n$ matrix defined as follows:

$$M^{(t)}(u,i) := \frac{\phi^*}{\max_j P_t^{(u)}(i,j)}.$$

Each row of the matrix $M^{(t)}$ corresponds to a possible next probe specification. We say that the stochastic vector q **violates** row u of $M^{(t)}$ if there exists $1 \le i \le n$, such that $M^{(t)}(u,i) < q_i$. Note that if row u of $M^{(t)}$ is violated by q, then according to (2), the next probe specification cannot be $P_t^{(u)}$.

Let $r_t := \sqrt{5t^*\phi^* sn \ln N_t}$. We say that a row u of $M^{(t)}$ is **good** if there exists $R \subseteq \{1,2,\ldots,n\}$ such that $|R| = r_t$ and $\sum_{i \in R} M^{(t)}(u,i) \le \phi^* s$.

We claim that if a row u is not good, then for the corresponding $P^{(u)}$, it holds that

$$\sum_{j=1}^{s} \max_{1 \le i \le n} P_t^{(u)}(i,j) \le r_t. \tag{4}$$

The proof is as follows: If a row u of $M^{(t)}$ is not good, then by definition, for any R of size r_t, $\sum_{i \in R} M^{(t)}(u,i) > \phi^* s$, thus for any R' that $\sum_{i \in R'} \frac{1}{\max_j P_t^{(u)}(i,j)} \le s$, it must hold that $|R'| < r_t$, therefore due to Lemma 16, it holds that $\sum_{j=1}^{s} \max_{1 \le i \le n} P_t^{(u)}(x_i,j) \le r_t$.

Due to (4) and (3), the amount of information brought by a set of probes $P_t^{(u)}$ where u is a bad row in $M^{(t)}$, is bounded by br_t bits. We show by an adversary argument that there exists a q that $\mathcal{A}''$ always choose probes corresponding to bad rows. At each round t, the adversary always chooses some q that violates all the good rows in $M^{(t)}$. According to Lemma 15, the adversary can do so as long as $t \le t^*$. Setting $\epsilon = \frac{1}{t^*}$ and $\delta = \phi^* s$ in Lemma 15, in each round, the adversary can increase the value of some q_i so that $\sum_{i=1}^{n} q_i$ is increased by at most $\frac{1}{t^*}$, thereby violating all good rows in the current $M^{(t)}$. Thus before round t^*, the vector q is always stochastic. Note that increasing the value of q_i will never make a violated row non-violated, so it will not make the adversary inconsistent.

Against such an adversary, at each round t, $\mathcal{A}''$ can only choose a probe specification $P_t^{(u)}$ where u is a bad row in $M^{(t)}$, according to Claim (4), which implies that

$$\sum_{j=1}^{s} \max_{1 \le i \le n} P_t(i,j) \le r_t = \sqrt{5t^*\phi^* sn \ln N_t} = \sqrt{5t^*\phi^* snC_{t-1} \ln 2}.$$

Due to (3), it holds that

$$\mathrm{E}[C_t] \le b \cdot \sum_j \max_i P_t(i,j) \le \sqrt{(5 \ln 2) b^2 t^* \phi^* snC_{t-1}},$$

where the expectation is conditioned on all previous rounds of communication. Therefore the following recursion holds

for the sequence of random variables $C_1, C_2, \ldots, C_t$:

$$\mathrm{E}[C_t \mid C_{t-1}] \leq \sqrt{(5\ln 2)b^2t^*\phi^*snC_{t-1}}\,.$$

The square root function is concave, thus by Jensen's inequality, it holds for the unconditional expectation that

$$\begin{aligned}\mathrm{E}[C_t] &= \mathrm{E}[\mathrm{E}[C_t \mid C_{t-1}]] \\ &\leq \mathrm{E}\left[\sqrt{(5\ln 2)b^2t^*\phi^*snC_{t-1}}\right] \\ &\leq \sqrt{(5\ln 2)b^2t^*\phi^*sn\cdot \mathrm{E}[C_{t-1}]}\,.\end{aligned}$$

Before the first probe, q is unknown to $\mathcal{A}''$, thus due to (2), for any i, j, $P_1(x_i, j) \leq \phi^*$, therefore

$$\mathrm{E}[C_1] \leq b\cdot\sum_j \max_i P_1(x_i, j) \leq b\phi^* s.$$

Let $a_1 := b\phi^* s$ and $a := (5\ln 2)b^2t^*\phi^*sn$. The following recursion holds for $\mathrm{E}[C_t]$ that

$$\begin{aligned}\mathrm{E}[C_1] &\leq a_1; \\ \mathrm{E}[C_t] &\leq (a\cdot \mathrm{E}[C_{t-1}])^{\frac{1}{2}}.\end{aligned}$$

By induction, $\mathrm{E}[C_t] \leq a_1^{2^{1-t}} a^{1-2^{1-t}}$.

After t^* rounds, the expected total number of bits received by $\mathcal{A}''$ is at least $n\cdot 2^{-2t^*}$, therefore

$$n\cdot 2^{-2t^*} \leq \sum_{t\leq t^*} \mathrm{E}[C_t] \leq \sum_{t\leq t^*} a_1^{2^{1-t}} a^{1-2^{1-t}} \leq a_1 a^{1-2^{-t^*}}.$$

With the assumption that $b \leq \mathrm{Polylog}(n)$ and $\phi^* \leq \frac{\mathrm{Polylog}(n)}{s}$, it holds that $a_1 \leq \mathrm{Polylog}(n)$ and $a \leq n\cdot\mathrm{Polylog}(n)$. Solving the above inequality, we have that $t^* \geq \log\log n - o(\log\log n) = \Omega(\log\log n)$. Theorem 13 is proved. ∎

4. CONCLUSION

In this paper, we propose to study the memory contention caused by concurrent data structure queries. To study the problem, we introduce a measure of contention to the classic cell-probe model of static data structures. We show that if all positive queries are equally probable and similarly all negatively are equally probable, then there exists a static dictionary which answers membership queries with asymptotically optimal performance of time, space and contention. For the general case that the query distribution is arbitrary, we show that for all data structure problems with non-degenerating VC-dimensions, if the randomness is used only for balancing the probes, then even with unbounded space, the time and contention cannot be both optimal.

A possible future direction is to remove the assumption of independent cell-probes in the lower bound. Note that we only rely on this assumption to make sure that the contention constraint of (2) holds conditioning on any particular sequence of previous cell-probes, which is required by the adversary argument. We suspect that with a more careful analysis, this assumption can be removed, which would imply that the lower bound holds not only for the randomized data structures that use the randomness only for balancing probes, but also for the true randomized data structures where the randomness is also involved in the computation of queries.

Another interesting and perhaps more realistic future direction is to study the contention caused by the *updates* in dynamic data structures.

5. REFERENCES

[1] J. Carter and M. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences (JCSS)*, 18(2):143–154, 1979.

[2] D. Culler, R. Karp, D. Patterson, A. Sahay, K. Schauser, E. Santos, R. Subramonian, and T. Von Eicken. LogP: Towards a realistic model of parallel computation. *ACM SIGPLAN Notices*, 28(7):1–12, 1993.

[3] M. Dietzfelbinger and F. Meyer auf der Heide. An optimal parallel dictionary. In *Proceedings of the first annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 360–368. ACM New York, NY, USA, 1989.

[4] M. Dietzfelbinger and F. Meyer auf der Heide. A new universal class of hash functions and dynamic hashing in real time. In *Proceedings of the 17th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 443, pages 6–19. Springer, 1990.

[5] M. Dietzfelbinger and F. Meyer auf der Heide. How to distribute a dictionary in a complete network. In *Proceedings of the twenty-second annual ACM Symposium on Theory of Computing (STOC)*, pages 117–127. ACM New York, NY, USA, 1990.

[6] C. Dwork, M. Herlihy, and O. Waarts. Contention in shared memory algorithms. *Journal of the ACM (JACM)*, 44(6):779–805, 1997.

[7] S. Fortune and J. Wyllie. Parallelism in random access machines. In *Proceedings of the tenth annual ACM Symposium on Theory of Computing (STOC)*, pages 114–118. ACM New York, NY, USA, 1978.

[8] M. Fredman, J. Komlós, and E. Szemerédi. Storing a Sparse Table with O(1) Worst Case Access Time. *Journal of the ACM (JACM)*, 31(3):538–544, 1984.

[9] M. Herlihy, B. Lim, and N. Shavit. Low contention load balancing on large-scale multiprocessors. In *Proceedings of the fourth annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 219–227. ACM New York, NY, USA, 1992.

[10] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[11] C. P. Kruskal, L. Rudolph, and M. Snir. A complexity theory of efficient parallel algorithms. *Theor. Comput. Sci.*, 71(1):95–132, 1990.

[12] R. Pagh and F. Rodler. Cuckoo hashing. *Journal of Algorithms*, 51(2):122–144, 2004.

[13] P. Tzeng and D. Lawrie. Distributing hot-spot addressing in large-scale multiprocessors. *IEEE Transactions on Computers*, 100(36):388–395, 1987.

[14] V. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

APPENDIX

A. PROOF OF LEMMA 14

The lemma is proved by first simulating the cell-probing algorithm $\mathcal{A}$ by a modified cell-probing algorithm $\mathcal{A}'$ which independently probes all cells in the table in every step, and

then running n instances of $\mathcal{A}'$ in parallel as a new cell-probing algorithm $\mathcal{A}''$.

First, we observe that a randomized cell-probe can be simulated with bounded error by independently probing all cells.

DEFINITION 17. *A product-space cell-probe to a table of s cells is a random set $J \in 2^{[s]}$ such that the probability space of J is a product probability space.*

For the rest of the proof, we assume the assumption of Theorem 13.

ASSUMPTION 18. *Let f be a data structure problem with VC-dim$(f) = n$. There exists a cell-probing scheme $(T, \mathcal{A})$ as described in Definition 12, such that $(T, \mathcal{A})$ solves f with performance parameters (s, b, t^*, ϕ^*), where the size of cell $b \leq \text{Polylog}(n)$ and the maximum contention $\phi^* \leq \frac{\text{Polylog}(n)}{s}$.*

LEMMA 19. *If Assumption 18 is true, then there exists a product-space cell-probing algorithm $\mathcal{A}'$, such that on any valid table $T_{S,q}$, for any query $x \in Q$, the following properties hold for the sequence of product-space cell-probes*

$$J_x^{(1)}, J_x^{(2)}, \ldots, J_x^{(t^*)}.$$

1. *At any step $t \leq t^*$, $\mathcal{A}'$ fails (returns a special symbol $\perp$) independently with probability at most $\frac{3}{4}$. Conditioned on that there is no failure after t^* steps, which is an event with probability at least 2^{-2t^*}, it holds that $J_x^{(1)}, J_x^{(2)}, \ldots, J_x^{(t^*)}$ are jointly independent and $\mathcal{A}'$ returns what $\mathcal{A}$ returns.*
2. *For any $t \leq t^*$, the total probability of each product-space cell-probe of $\mathcal{A}'$*

$$\sum_{j \in [s]} \Pr\left[j \in J_x^{(t)}\right] \leq 1; \tag{5}$$

3. *For any $t \leq t^*$, the contention of any cell $j \in [s]$*

$$q(x_i) \cdot \Pr\left[j \in J_x^{(t)}\right] \leq \phi^*. \tag{6}$$

PROOF. A cell-probe of $\mathcal{A}$ can be represented as a random variable $I \in [s]$, where I denotes the probed cell. Let $p_i := \Pr[I = i]$. Let $J \in 2^{[s]}$ represent a product-space cell-probe. Given a probability vector p, a cell-probe I is simulated by a product-space cell-probe as follows: Independently probe each cell $i \in [s]$ with probability $p'_i := \min\{p_i, \frac{1}{2}\}$. The resulting set is J. If $|J| \neq 1$, then fails; if $J = \{i\}$, then fails with a probability $\epsilon_i := \min\{p_i, 1 - p_i\}$. Let $I = i$ if not fail.

Case 1: $p_i \leq \frac{1}{2}$ for all $i \in [s]$. Then for all $i \in [s]$, $p'_i = p_i$ and $\epsilon_i = p_i$. Let $\rho = \prod_{j \in [s]}(1 - p_j)$. Since $p_i \leq \frac{1}{2}$ for all $i \in [s]$, it holds that $\rho \geq \frac{1}{4}$.
The probability

$$\begin{aligned}\Pr[I = i] &= (1 - \epsilon_i) \cdot \Pr[J = \{i\}] \\ &= (1 - p_i) \cdot p_i \prod_{j \neq i}(1 - p_j) \\ &= p_i \rho,\end{aligned}$$

which is proportional to p_i. The procedure succeeds with probability $\rho \geq \frac{1}{4}$.

Case 2: Let $p_0 > \frac{1}{2}$ and all other $p_i < \frac{1}{2}$. Then $p'_0 = \frac{1}{2}$ and $\epsilon'_0 = 1 - p_0$, and for all $i > 0$, it holds that $p'_i = p_i$ and $\epsilon'_i = p_i$. Let $\rho' = \prod_{j>0}(1 - p_j)$. It holds that $\rho' > \frac{1}{2}$ since $\sum_{j>0} p_j = 1 - p_0 < \frac{1}{2}$.
For $i \neq 0$,

$$\Pr[I = i] = (1 - p_i) \cdot \Pr[J = \{i\}] = \frac{1}{2}\rho' p_i;$$

and for cell 0,

$$\Pr[I = 0] = p_0 \cdot \Pr[J = \{i\}] = \frac{1}{2}\rho' p_0.$$

The procedure succeeds with probability $\frac{1}{2}\rho' > \frac{1}{4}$.

For both cases, a cell-probe of $\mathcal{A}$ is simulated by a product-space cell-probe with a probability at least $\frac{1}{4}$. The event of a failure occurs independently with probability at most $\frac{3}{4}$. With a probability at least 2^{-2t^*}, no failure occurs at all, conditioned on which its is obvious that $\mathcal{A}'$ can simulate $\mathcal{A}$, and the product-space cell-probes are jointly independent since the cell-probes of $\mathcal{A}$ are jointly independent.

The total probability of a product-space cell-probe is

$$\sum_i \Pr[i \in J] = \sum_i p'_i \leq 1.$$

The probability of a probe to each cell is no greater than before, therefore the maximum contention ϕ^* not increased. □

We observe that by running n instances of the product-space cell-probing algorithm in parallel, the behavior of each individual instance depend only on the marginal distribution of cell-probes of the instance, but does not depend on the joint distribution of cell-probes of all n instances. Thus, the joint distribution of cell-probes can be arbitrarily chosen by us as long as the marginal distribution of cell-probes of each individual instance is the same as before.

LEMMA 20. *Let $\mathcal{A}''$ be an algorithm that on a valid table $T_{S,q}$, for a set of n queries $\{x_1, x_2, \ldots, x_n\} \in \binom{Q}{n}$, at step t, $\mathcal{A}''$ randomly probes n sets of cells $(L_{x_1}^{(t)}, L_{x_2}^{(t)}, \ldots, L_{x_n}^{(t)})$. If for every x_i where $1 \leq i \leq n$ and every $t \leq t^*$, the marginal distribution of $L_{x_i}^{(t)}$ is identical to the distribution of $J_{x_i}^{(t)}$, where $J_{x_i}^{(t)}$ is the t's product-space cell-probe of the algorithm $\mathcal{A}'$ on the same table $T_{S,q}$, then $\mathcal{A}''$ returns $f(x_i, S)$ for expected $n \cdot 2^{-2t^*}$ number of x_i.*

PROOF. Let $\mathcal{A}''$ run an instance of $\mathcal{A}'$ with input x_i in parallel for every x_i, denoted as $\mathcal{A}'_{x_i}$. Let the set of cells probed by each individual $\mathcal{A}'_{x_i}$ at time t be $L_{x_i}^{(t)}$. On a fixed table $T_{S,q}$ and for a fixed x_i, since $L_{x_i}^{(t)}$ is identically distributed as $J_{x_i}^{(t)}$, every individual instance of $\mathcal{A}'_{x_i}$ simulates a running instance of $\mathcal{A}'$ with input query x_i. By Lemma 19, each $\mathcal{A}'_{x_i}$ terminates in t^* steps without failure with probability at least 2^{-2t^*}, thus by the linearity of expectation, after t^* time, the expected total number of terminated instances is at least $n \cdot 2^{-2t^*}$. □

In the next lemma, we construct a joint distribution of cell-probes which minimizes the expected total number of probed cells.

LEMMA 21. *For any probability distribution of $J_i \subseteq [s]$ where $1 \leq i \leq n$ and each J_i is chosen from a product probability space, there exists a joint distribution $(L_1, L_2, \ldots, L_n)$,*

such that for every $1 \le i \le n$, L_i *is identically distributed as* J_i*, and it holds that*

$$\mathrm{E}\left[\left|\bigcup_{1\le i\le n} L_i\right|\right] \le \sum_{j\in[s]} \max_{1\le i\le n} \Pr\left[j \in J_i\right].$$

PROOF. We construct the joint distribution of $(L_i)_i \in (2^{[s]})^n$ as follow.

- Let $\tilde{p}_j = \max_{1\le i\le n} \Pr[j \in J_i]$. Choose each $j \in [s]$ independently with probability $\tilde{p}_j$. Let B denote the set of chosen elements of $[s]$.
- For every $1 \le i \le n$, let each $j \in B$ join L_i independently with probability $\frac{\Pr[j\in J_i]}{\tilde{p}_j}$. Note that $\tilde{p}_j \ge \Pr[j \in J_i]$, therefore the probability is well-defined.

For each $1 \le i \le n$, and for every $j \in [s]$, j joins L_i independently with probability $\tilde{p}_j \cdot \Pr[j \text{ is chosen to } L_i \mid j \in B] = \Pr[j \in J_i]$, thus each L_i is identically distributed as J_i.

Note that for every L_i, all of its elements are chosen from set B. It holds that

$$\mathrm{E}\left[\left|\bigcup_{1\le i\le n} L_i\right|\right] \le \mathrm{E}\left[|B|\right] = \sum_{j\in[s]} \tilde{p}_j = \sum_{j\in[s]} \max_{1\le i\le n} \Pr[j \in J_i].$$

□

Proof of Lemma 14: Let $\{x_1, x_2, \ldots, x_n\} \in \binom{Q}{n}$ be a set of queries which achieves the VC-dimension VC-dim$(f) = n$, i.e. any Boolean assignment of $f(x_i, S)$ is possible. Let such $\{x_1, x_2, \ldots, x_n\}$ be the input query set to $\mathcal{A}''$, where $\mathcal{A}''$ is as described in Lemma 20. By an information theoretical argument, in the worst case, $\mathcal{A}''$ has to collect expected $n \cdot 2^{-2t^*}$ bits information after t^* steps. Let the joint distribution of $(L^{(t)}_{x_1}, L^{(t)}_{x_2}, \ldots, L^{(t)}_{x_n})$ of $\mathcal{A}''$ be constructed as in Lemma 21, the total number of cells probed by $\mathcal{A}''$ in step t with $(L^{(t)}_{x_1}, L^{(t)}_{x_2}, \ldots, L^{(t)}_{x_n})$ is bounded. Let the $n \times s$ matrix P_t defined as $P_t(i,j) := \Pr[j \in L^{(t)}_{x_i}]$, and let $q_i := q(x_i)$. Since each $L^{(t)}_{x_i}$ is identically distributed as $J^{(t)}_{x_i}$ of $\mathcal{A}'$ which is described in Lemma 19, due to (2) and (3), it holds for the P_t that $\sum_{j\in[s]} P_t(i,j) \le 1$ and $\max_{j\in[s]} P_t(i,j) \le \frac{\phi^*}{q_i}$. Due to Lemma 21, the expected number of bits collected by $\mathcal{A}''$ in step t is bounded by $b \cdot \sum_{j\in[s]} \max_{1\le i\le n} P_t(i,j)$. By seeing the running instance of the algorithm $\mathcal{A}''$ with the input $\{x_1, x_2, \ldots, x_n\}$ as the player $\mathcal{A}''$ of the communication game, and the table $T_{S,q}$ as the black-box $\mathcal{B}$ with private input q, Lemma 14 is proved. ∎