Should Tables be Sorted: Cheat Sheet

William Gasarch
Clyde Kruskal

April 19, 2020

The Cell Probe Model

Definition The *Cell Probe Model* for search is as follows:

1. The size of the universe is U. The universe is $\{1, \ldots, U\}$.
2. The number of elements from the universe that we will store is n.
3. The function *PUT* takes $A \in \binom{[U]}{n}$ and outputs the elements of A in some order. This tells us how to store A in an array.
4. An algorithm *FIND* that, on input $x \in U$, probes the array (by asking 'What is in cell c'), and based on the answer probes another cell, etc, and then says either x is in A, or x is not in A.

Examples One: Sort

- The function *PUT* takes $A \in \binom{[U]}{n}$ and puts them in an n-array SORTED.
- The algorithm *FIND* does Binary Search.

Number of Probes $\lceil (\rceil \log(n+1))$.
Can we do better?

0 Probes But Its Stupid

Silly Example: $U = n$.

- The function *PUT* takes $A \in \binom{[n]}{n}$ and puts A into an n-array. Note that *everything in U is in the table*.
- Just say YES, since EVERY element is in the table.

Number of Probes 0.

Caveat The Model only asked us to determine if x is IN the table, not to find WHERE in the table x is.

1 Probes But Its Stupid

Silly Example: $U = n + 1$.

- The function PUT takes $A \in \binom{[n+1]}{n}$, notes that z is the ONLY element of $U - A$, and puts $z - 1 \pmod{U}$ into the first spot of the array.
- Given x, look at the first spot of the array and you see w. If $x = w + 1 \pmod{U}$ then say NO, else say YES.

Number of Probes 1.

1 Probes But Its HW

$U = 2n - 2$.
I have notes on this and there will be a HW on it.

Main Result

We saw that if U is not that big then we can do FIND with $<< \log n$ probes.

Main Result

We saw that if U is not that big then we can do FIND with $<<\log n$ probes.
The main result is that if U is BIG then it REQUIRES $\log n$ probes.

Lemma on Sorting

Lemma If $U \geq 2n-1$ and the elements are always put in in sorted order than ANY probe algorithm requires $\geq \log(n+1)$ probes.

Lemma on Sorting

Lemma If $U \geq 2n - 1$ and the elements are always put in in sorted order than ANY probe algorithm requires $\geq \log(n+1)$ probes.
We omit the proof. Its in the paper. It is an adversary argument.

Lemma on Sorting

Lemma If $U \geq 2n - 1$ and the elements are always put in in sorted order than ANY probe algorithm requires $\geq \log(n+1)$ probes.
We omit the proof. Its in the paper. It is an adversary argument.
We can rephrase the lemma as follows:
Lemma Let σ be the permutation $(1, 2, 3, \ldots, n)$. If $U \geq 2n - 1$ and the elements are always put in in the array using the perm σ then ANY probe algorithm requires $\geq \log(n+1)$ probes.

Lemma on Any Permutation

Let $\sigma = (3, 4, 5, 1, 2)$.
Then we can think of putting elements into an array using this σ.
$A[1]$ would have the 3rd largest elements
$A[2]$ would have the 4th largest elements
$A[3]$ would have the 5th largest elements
$A[4]$ would have the 1st largest elements
$A[5]$ would have the 2nd largest elements
Lemma Let σ be any permutation of $\{1, \ldots, n\}$. If $U \geq 2n - 1$ and the elements are always put in in the array using the perm σ then ANY probe algorithm requires $\geq \log(n + 1)$ probes.
We omit the proof. Its in the paper. It is an adversary argument.

Main Theorem

Theorem Let $U \geq R_n(2n-1, n!)$ (n-ary Ramsey, $2n-1$ homog set, $n!$ color). Then any Cell Probe Search Algorithm requires $\log_2(n+1)$ probes.

Main Theorem

Theorem Let $U \ge R_n(2n-1, n!)$ (n-ary Ramsey, $2n-1$ homog set, $n!$ color). Then any Cell Probe Search Algorithm requires $\log_2(n+1)$ probes.

Proof Color $\binom{[U]}{n}$ as follows: Color $X \in \binom{[U]}{n}$ by σ such that X was put into the array via σ.
By the n-ary Ramsey Theorem and the definition of U there exists $2n-1$ element that are always put into the array using the SAME perm, which we call σ.
By Lemma above, if you restrict the cell probe algorithm to there $2n-1$ elements then ANY probe-algorithm requires $\log_2(n+1)$ probes.