Cell Probe Lower Bounds and Approximations for Range Mode

Mark Greve
Aarhus University

Jakob Truelsen
Aarhus University


MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS

The Range Mode Problem

Preprocess an array A of n elements into a space efficient data structure supporting the following queries.

Mode(i, j)

Find a most occurring element in A[i..j].

	0	1	2 (i)	3	4	5	6	7	8	9	10	11 (j)	12	13
A	a	b	a	e	c	d	c	d	c	b	b	a	x	t

mode(2, 11) = c

Approximate mode(i,j)

Find an element occurring at least 1/(1+ε) times as often as a mode in A[i..j].

Applications

The mode is a general statistical measure. The range mode applies among other things to describe discrete events. For instance if we create a list over the winners of football matches, a query would correspond to asking between these two matches, who won the most matches.

Selected Previous Results

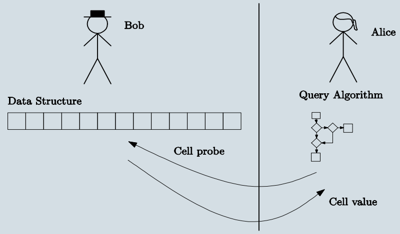
- Previously no non-trivial lower bound has been shown for neither the range mode nor the approximate range mode problem.
- Grabowski and Petersen [2] have presented a data structure supporting range mode queries in constant time and using $O(n^2 \log\log n / \log^2 n)$ words of space.
- Petersen [4] has presented a data structure supporting range mode queries in $O(n^{\varepsilon})$ time and using $O(n^{2-2\varepsilon})$ words of space.
- Bose, Kranakis, Morin and Tang [3] have presented a data structure supporting (1+ε)-approximate range mode problem queries in $O(\log\log n + \log \varepsilon^{-1})$ time using linear space.

Cell Probe Lower Bounds for the Range Mode Problem

Using communication complexity techniques, we prove a query lower bound for the range mode problem of $\Omega(\log n / \log (Sw/n))$ time for any data structure using S cells of space. This means that any data structure using $O(n \log^{O(1)} n)$ space will require a query time of $\Omega(\log n / \log\log n)$, and that for constant time queries we require $n^{1+\Omega(1)}$ words of space.

The proof uses techniques introduced by Miltersen, and later refinements [5]. Here Alice simulates the query algorithm and Bob holds the data structure, whenever Alice queries a cell, Bob responds with the contents.

Pătraşcu showed a lower bound for lopsided set disjointness. We show how to solve such instances using a range mode data structure. Thus obtaining the claimed query time/space tradeoff.



(1+ε)-approximate Range Mode

We constructed a very simple data structure using a few bit tricks that answers 3-approximate range mode queries in constant time using linear space. The best previous approximation achieving these bounds was a 4-approximation by Bose, Kranakis, Morin and Tang [3].

Building upon any of these data structures, we can, given an arbitrary ε, construct a data structure that answers (1+ε)-approximate range mode queries in $O(\log \varepsilon^{-1})$ time using $O(n/\varepsilon)$ space. This construction is similar to [3].

References

[1] Greve, Jørgensen, Larsen, Truelsen. *Cell probe lower bounds and approximations for range mode.* Manuscript. 2010.

[2] Grabowski, Petersen. *Range mode and range median queries in constant time and sub-quadratic space*. Inf. Process. Lett. 2008.

[3] Bose, Kranakis, Morin, Tang. *Approximate range mode and range median queries.* STACS 2005.

[4] Petersen. *Improved bounds for range mode and range median queries.* CCTTPCS 2008.

[5] Pătraşcu, Thorup. *Higher lower bounds for near-neighbor and further rich problems.* FOCS 2006.

The Range k-frequency Problem

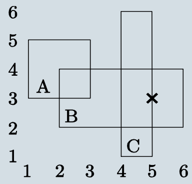
Preprocess an array A of n elements into a space efficient data structure supporting the following queries.

Range k-frequency(i, j)

Find an element in A[i..j] occurring exactly k times.

We prove that for fixed k>1, this problem is equivalent to the 2D orthogonal rectangle stabbing problem. And for k=1 it is no harder than 4-sided 3D orthogonal range emptiness.

The following is an example of the reduction from 2D orthogonal rectangle stabbing to range 2-frequency. The rectangles are projected on the the x- and y-axis. We visit the end points of the line segments in increasing order, adding starting points once and ending points twice. Finally the two strings X and Y are concatenated to form the array A.



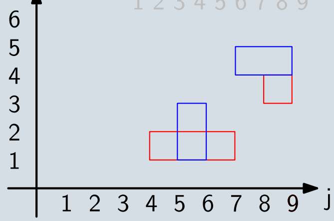
$\Rightarrow$

X=[(6,B)(6,B)(5,C)(5,C)(4,C)(3,A)(3,A)(2,B)(1,A)]
Y=[(1,C)(2,B)(3,A)(4,B)(4,B)(5,A)(5,A)(6,C)(6,C)]

$$A = [\underbrace{BBCCCAABA}_{X}\underbrace{CBABBAACC}_{Y}]$$

Below is an example of the reduction from range 2-frequency to 2D orthogonal rectangle stabbing. Notice that "a" has frequency two when 1 ≤ i ≤ 2 and 4 ≤ j ≤ 7. This gives rise to the bottommost red rectangle. In a similar way all other ranges where a or b has frequency two give rise to the remaining rectangles. Thus if point (i,j) stabs a rectangle, then there is an element with frequency two in A[i..j].

A = [c a b a b d b a e]
1 2 3 4 5 6 7 8 9



MADALGO – Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation