Theoretical Computer Science 647 (2016) 59–73

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Dynamic range majority data structures ☆

Amr Elmasry [a], Meng He [b], J. Ian Munro [c], Patrick K. Nicholson [d,*]

[a] Department of Computer Engineering and Systems, Alexandria University, Egypt
[b] Faculty of Computer Science, Dalhousie University, Canada
[c] David R. Cheriton School of Computer Science, University of Waterloo, Canada
[d] Max-Planck-Institut für Informatik, Saarbrücken, Germany

ARTICLE INFO

Article history:
Received 4 October 2014
Received in revised form 26 December 2015
Accepted 29 July 2016
Available online 9 August 2016
Communicated by G.F. Italiano

Keywords:
Data structures
Range queries
Range majority
Heavy hitters

ABSTRACT

Given a set $\mathcal{P}$ of n coloured points on the real line, we study the problem of answering range α-majority (or "heavy hitter") queries on $\mathcal{P}$. More specifically, for a query range $\mathcal{Q}$, we want to return each colour that is assigned to more than an α-fraction of the points contained in $\mathcal{Q}$. We present a new data structure for answering range α-majority queries on a dynamic set of points, where $\alpha \in (0, 1)$. Our data structure uses $O(n)$ space, supports queries in $O((\lg n)/\alpha)$ time, and updates in $O((\lg n)/\alpha)$ amortized time. If the coordinates of the points are integers, then the query time can be improved to $O(\lg n/(\alpha \lg\lg n))$. For constant values of α, this improved query time matches an existing lower bound, for any data structure with polylogarithmic update time. We also generalize our data structure to handle sets of points in d dimensions, for $d \geq 2$, as well as dynamic arrays, in which each entry is a colour.

1. Introduction

Many problems in computational geometry deal with point sets that have information encoded as colours assigned to the points. In this paper, we design dynamic data structures for the *range α-majority problem*, in which we want to report colours that appear *frequently* within an axis-aligned query rectangle. This problem is useful in database applications in which we would like to know typical attributes of the data points in a query range [22,23]. For the one-dimensional case, where the points represent time stamps, this problem has data mining applications for network traffic logs, similar to those of coloured range counting (cf. [16]).

Formally, we are given a set, $\mathcal{P}$, of n points, where each point $p \in \mathcal{P}$ is assigned a colour c from a set, C, of colours. The colour of p is denoted by $\mathrm{col}(p) = c$. We are also given a fixed parameter $\alpha \in (0, 1)$ that defines the threshold for determining whether a colour is to be considered frequent. Our goal is to design a *dynamic range α-majority data structure* that can perform the following operations:

- QUERY($\mathcal{Q}$): We are given an axis-aligned hyper-rectangle $\mathcal{Q}$ as a query. Let $\mathcal{P}(\mathcal{Q})$ be the set $\{p \mid p \in \mathcal{P} \cap \mathcal{Q}\}$, and $\mathcal{P}(\mathcal{Q}, c)$ be the set $\{p \mid p \in \mathcal{P}(\mathcal{Q}), \mathrm{col}(p) = c\}$. The answer to the query $\mathcal{Q}$ is the set of colours $C^\star$ such that for each colour $c \in C^\star$, $|\mathcal{P}(\mathcal{Q}, c)| > \alpha|\mathcal{P}(\mathcal{Q})|$, and for all $c \notin C^\star$, $|\mathcal{P}(\mathcal{Q}, c)| \leq \alpha|\mathcal{P}(\mathcal{Q})|$. We refer to a colour $c \in C^\star$ as an *α-majority*

☆ A preliminary version of this work [13] appeared in the 22nd International Symposium on Algorithms and Computation (ISAAC 2011).

* Corresponding author at: Nokia Bell Labs, Blanchardstown Business and Technology Park, Snugborough Road, Blanchardstown, Dublin 15, Ireland.
E-mail addresses: elmasry@diku.dk (A. Elmasry), mhe@cs.dal.ca (M. He), imunro@cs.uwaterloo.ca (J.I. Munro), pnichols@mpi-inf.mpg.de (P.K. Nicholson).

http://dx.doi.org/10.1016/j.tcs.2016.07.039
0304-3975/

for $\mathcal{Q}$, and this type of query as an α-*majority query*. When $\alpha = 1/2$, the problem is to identify the majority colour in $\mathcal{Q}$, if such a colour exists.
- INSERT(p, c): Insert a point p with colour c into $\mathcal{P}$.
- DELETE(p): Remove the point p from $\mathcal{P}$.

1.1. Related work

For a comprehensive survey of similar types of *range queries* see the recent survey by Skala [30]; we mention only the most closely related results here. In the following we note that the threshold $\alpha \in (0, 1)$ is *fixed*—but not necessarily constant—at construction time. For some results, which we further distinguish by calling *parameterized*, some $\beta \in (\alpha, 1)$ is given at query time and the query is expected to return the β-majorities in the query range. If we do not specify the value of β, the reader may assume the result is for the fixed case: i.e., $\beta = \alpha$ for all queries.

1.1.1. Static case

Karpinski and Nekrich [22] first studied the problem of answering range α-majority queries, which they called *coloured α-domination* queries. In the *static case*, they gave an $O(n/\alpha)$ space data structure that supports one-dimensional queries in $O((\lg n \lg\lg n)/\alpha)$ time,[1] and an $O((n \lg\lg n)/\alpha)$ space data structure that supports queries in $O((\lg n)/\alpha)$ time. For points in d dimensions, for constant $d \geq 2$, they gave a static $O((n \lg^{d-1} n)/\alpha)$ space data structure that supports queries in $O((\lg^d n)/\alpha)$ time.

Durocher et al. [11] gave a static data structure occupying $O(n(\lg(1/\alpha) + 1))$ space and answers range α-majority queries in an array in $O(1/\alpha)$ time. The array setting is equivalent to the point set having x-coordinates $1, \ldots, n$. Their data structure is based on the idea that it is possible to produce a short list of candidate α-majorities for any query, and then efficiently verify the frequencies of these candidates using succinct data structures. In the journal version of the same paper [12], they described how to extend their technique to d dimensions for constant $d \geq 2$, resulting in an $O(n \lg^{d-1} n)$ space data structure that supports range α-majority queries in $O(\lg^d n/\alpha)$ time.

Gagie et al. [15] considered the problem in 2D arrays, and also improved the static one-dimensional result to $O(n(\min(\lg(1/\alpha), H) + 1))$ space, where $H \leq \lg n$ is the zeroth-order empirical entropy of the sequence stored in the array. The same authors also described how to improve the query time to $O(1/\beta)$ for the parameterized case. A similar result was proved independently by Chan et al. [7]. Recently, all these results were improved even further by Belazzougui et al. [4] who gave an $O(n)$ space data structure with $O(1/\alpha)$ query time for the static non-parameterized case, and an $O(n \lg\lg n)$ space data structure with $O(1/\beta)$ query time for the static parameterized case. They also presented several space-time trade offs for the parameterized case.

For the two-dimensional static case, Wilkinson [33] presented an improved data structure that occupies $O(n \lg^{\varepsilon} n(\lg(1/\alpha) + 1))$ space, for any constant $\varepsilon > 0$, and can answer queries in $O(\lg n/\alpha)$ time. To get this speedup Wilkinson observed that we can break the query into three phases. In the first phase, we extract a list of candidates from our data structure. In the second phase, we use approximate range counting to reduce the size of the candidate list to $O(1/\alpha)$: i.e., we eliminate a lot of candidates before spending time verifying their frequencies exactly. Then, finally, in the third phase we use an expensive exact range counting query to determine which of the candidates from this short list are actually α-majorities.

Navarro and Russo [28]—also at ISAAC 2011—presented data structures for the two-dimensional case. A later final version of their paper [27] (with Y. Nekrich), provides several trade-offs with the previous results. Their results are stated in terms of several parameters, but we simplify them to be in terms of an $n \times n$ grid for simplicity. In particular, for the two-dimensional static case, they can answer α-majority queries in $O(n)$ space and $O(\lg^3 n/\alpha)$ time. However, for the larger space case their results fall short of Wilkinson's [33].

1.1.2. Dynamic case

In the dynamic case, Karpinski and Nekrich [22] gave an $O(n/\alpha)$ space data structure for one-dimensional queries that supports queries and insertions in $O((\lg^2 n)/\alpha)$ time, and deletions in $O((\lg^2 n)/\alpha)$ amortized time. They also gave an alternative $O((n \lg n)/\alpha)$ space data structure that supports queries and insertions in $O((\lg n)/\alpha)$ time, and deletions in $O((\lg n)/\alpha)$ amortized time. For points in d dimensions they gave a dynamic $O((n \lg^{d-1} n)/\alpha)$ space data structure that supports queries and insertions in $O((\lg^{d+1} n)/\alpha)$ time, and deletions in $O((\lg^{d+1} n)/\alpha)$ amortized time.

For the two-dimensional case, as in the static case, Navarro, Nekrich, and Russo [27] provide several trade-offs. In the two-dimensional dynamic case, their data structure occupies $O(n \lg n)$ space, and has $O((\lg^3 n)/(\alpha \lg\lg n))$ query time, and $O((\lg^3 n)/(\lg\lg n))$ updates (without amortization).

1.1.3. Approximate variants of the problem

Researchers have also examined an approximate version of the range α-majority problem, in which the solution must contain all the α-majorities in a query range, but can also contain some false positives. This is a data structure variant of the very heavily studied "heavy hitters" problem, in which we wish to find α-majorities (possibly allowing some false

[1] $\lg n$ denotes $\lceil \log_2 n \rceil$.