Compact Encodings and Indexes for the Nearest Larger Neighbor Problem

Seungbum Jo[1], Rajeev Raman[2], and Srinivasa Rao Satti[1]

[1] Seoul National University, Seoul, South Korea
sbcho@tcs.snu.ac.kr, ssrao@cse.snu.ac.kr
[2] University of Leicester, Leicester, UK
r.raman@leicester.ac.uk

Abstract. Given a d-dimensional array, for any integer $d > 0$, the *nearest larger value* (NLV) query returns the position of the element which is closest, in L_1 distance, to the query position, and is larger than the element at the query position. We consider the problem of preprocessing a given array, to construct a data structure that can answer NLV queries efficiently. In the 2-D case, given an $n \times n$ array A, we give an asymptotically optimal $O(n^2)$-bit encoding that answers NLV queries in $O(1)$ time. When A is a binary array, we describe a simpler $O(n^2)$-bit encoding that also supports NLV queries in $O(1)$ time. Using this, we obtain an index of size $O(n^2/c)$ bits that supports NLV queries in $O(c)$ time, for any parameter c, where $1 \le c \le n$, matching the lower bound. For the 1-D case we consider the *nearest larger right value* (NLRV) problem where the nearest larger value to the right is sought. For an array of length n, we obtain an index that takes $O((n/c)\log c)$ bits, and supports NLRV queries in $O(c)$ time, for any any parameter c, where $1 \le c \le n$, improving the earlier results of Fischer et al. and Jayapaul et al.

1 Introduction and Motivation

We consider cases of the following general problem. We are given a d-dimensional array A consisting of (possibly not all distinct) items from an ordered universe. After preprocessing A we are given a series of queries, each of which specifies an element of A, and our objective is to return the element of A nearest to the query element that is strictly larger than the query element. One may also restrict that the answer to the query comes from some particular sub-array (e.g. a quadrant) of A. Specifically, we consider the two queries below:

NLRV: Given a 1-D array A and an index i, returns the first larger element to i's right, i.e., returns $\min\{j > i | A[j] > A[i]\}$ (and is undefined if this set is empty). The query NLLV is defined analogously to i's left.

NLV: Given a d-dimensional array A and an index $p = (i_1, i_2 \ldots, i_d)$, returns an index $q = (i'_1, i'_2 \ldots, i'_d)$ such that $A[q] > A[p]$ and the distance between p and q, $dist(p, q) = |i_1 - i'_1| + |i_2 - i'_2| + \cdots + |i_d - i'_d|$, is minimized (note that we use the L_1 metric). In case of many equidistant larger values, ties can be broken arbitrarily. If there is no larger value, then it is undefined.

Encoding and indexing models. We consider these problems in two different models that have been studied in the succinct data structures literature, namely the *indexing* and *encoding* models. In both these models, the data structure is created after preprocessing A. In the indexing model, the queries can be answered by probing the data structure as well as the input data, whereas in the encoding model, the query algorithm cannot access the input data.

Previous Work and Motivation. The off-line version of this problem: given A, to compute nearest larger values for *all* entries of A (the ANLV problem), has been studied previously[1]. In the 1-D case, Berkman et al. [3] noted that the best highly-parallel solutions to a number of tasks including answering range minimum queries, triangulating monotone polygons and matching parentheses, are obtained by reducing to the ANLV problem, and efficient parallel solutions to the ANLV problem were also given by the same authors. A number of plausible applications, and algorithms, for the ANLV problem in 2 and higher dimensions were given by Asano et al. [1], and time-space tradeoffs for the 1-D case were given by Asano and Kirkpatrick [2].

Fischer et al. [10] considered the problem of supporting NLRV and NLLV in the 1-D case, and showed how a data structure supporting these two queries is essential to a space-efficient compressed suffix tree. They also considered the problem of supporting NLRV and NLLV in the indexing model, and gave a space-time tradeoff (the precise result is given later). Fischer [9] gave a structure in the encoding model that uses $2.54n + o(n)$ bits and supports NLRV and NLLV queries in $O(1)$ time.

Jayapaul et al. [12] considered the problem of encoding and indexing NLV in the 2-D case. Below, we describe the directly relevant results from their work.

Our results. We obtain new results for encoding and/or indexing 1-D and 2-D nearest larger value queries. In all the 2-D results we assume L_1 distances.

- We show that 2-D NLV can be encoded in the asymptotically optimal $O(n^2)$ bits in the *general* case. Jayapaul et al. showed this only for the case where all elements of A are distinct. Distinctness is a strong assumption in these kinds of problems. For example, in the 1-D case with distinct values, NLRV and NLLV can both be trivially encoded by the Cartesian tree (giving a $2n - O(\log n)$ bit encoding). By contrast, if we do not assume distinctness, the optimal space is about $2.54n$ bits, and the data structure achieving this bound is also more complex [9]. Also, Asano et al. [1] remark that the ANLV problem for any dimension is "simplified considerably" if one assumes distinctness. In fact, for the general case, Jayapaul et al. were only able to give an encoding with size $\Theta(n^2 \log\log n)$ bits and $O(1)$ query time.

[1] The terminology varies considerably. Berkman et al. studied the all nearest *smaller* values (ANSV) problem, which is symmetric to the ANLV problem. The previous/next smaller value (PSV/NSV) problems of Fischer et al. are symmetric to the NLLV/NLRV problems. Asano et al. and Jayapaul et al. call the NLV problem the *nearest larger neighbour* (NLN) problem: we consider the term "neighbour" to be mildly misleading, as the answer may not be a neighbour of the query element in A.