Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Encoding nearest larger values ☆

Michael Hoffmann [a], John Iacono [b], Patrick K. Nicholson [c,*], Rajeev Raman [a]

[a] University of Leicester, University Road, Leicester, United Kingdom
[b] New York University, School of Engineering, 6 MetroTech Center, Brooklyn, NY, United States
[c] Nokia Bell Labs, Clyde House, Blanchardstown Business and Technology Park, Dublin, Ireland

ARTICLE INFO

Article history:
Received 10 June 2016
Received in revised form 30 December 2016
Accepted 14 February 2017
Available online xxxx

Keywords:
Data structures
Encoding data structures
Succinct data structures

ABSTRACT

In *nearest larger value (NLV)* problems, we are given an array $A[1..n]$ of distinct numbers, and need to preprocess A to answer queries of the following form: given any index $i \in [1, n]$, return a "nearest" index j such that $A[j] > A[i]$. We consider the variant where the values in A are distinct, and we wish to return an index j such that $A[j] > A[i]$ and $|j - i|$ is minimized, the *nondirectional NLV (NNLV)* problem. We consider NNLV in the *encoding* model, where the array A is deleted after preprocessing.

The NNLV encoding problem turns out to have an unexpectedly rich structure: the *effective entropy* (optimal space usage) of the problem depends crucially on details in the definition of the problem. Of particular interest is the tiebreaking rule: if there exist two nearest indices j_1, j_2 such that $A[j_1] > A[i]$ and $A[j_2] > A[i]$ and $|j_1 - i| = |j_2 - i|$, then which index should be returned? For the tiebreaking rule where the rightmost (i.e., largest) index is returned, we encode a path-compressed representation of the Cartesian tree that can answer all NNLV queries in $1.89997n + o(n)$ bits, and can answer queries in $O(1)$ time. An alternative approach, based on *forbidden patterns*, achieves a very similar space bound for two tiebreaking rules (including the one where ties are broken to the right), and (for a more flexible tiebreaking rule) achieves $1.81211n + o(n)$ bits. Finally, we develop a fast method of counting distinguishable configurations for NNLV queries. Using this method, we prove a lower bound of $1.62309n - \Theta(1)$ bits of space for NNLV encodings for the tiebreaking rule where the rightmost index is returned.

1. Introduction

Nearest Larger Value (NLV) problems have had a long and storied history. Given an array $A[1..n]$ of values, the objective is to preprocess A to answer queries of the general form: given an index i, report the index or indices nearest to i that contain values strictly larger that $A[i]$. If no such index exists, then $A[i]$ is the maximum element in A, and we return -1.

Berkman et al. [1] studied the parallel pre-processing for this problem and noted a number of applications, such as parenthesis matching and triangulating monotone polygons. The connection to string algorithms for both the data structuring and the pre-processing variants of this problem is since well-established.

☆ A preliminary version appeared in the Proceedings of the 26th Annual Symposium on Combinatorial Pattern Matching (CPM 2015).

* Corresponding author.
E-mail addresses: mh55@leicester.ac.uk (M. Hoffmann), iacono@nyu.edu (J. Iacono), pat.nicholson@nokia.com (P.K. Nicholson), r.raman@leicester.ac.uk (R. Raman).

http://dx.doi.org/10.1016/j.tcs.2017.02.017

Table 1
Number of distinguishable configurations of nearest larger value problems with various tiebreaking rules.

n	1	2	3	4	5	6	7	8	9	10	11	12
rule I	1	2	5	14	40	116	341	1010	3009	9012	27087	81658
rule II	1	2	5	14	42	126	383	1178	3640	11316	35263	110376
rule III	1	2	5	12	32	88	248	702	1998	5696	16304	46718

Since the definition of "nearest" is a bit ambiguous, we propose replacing it by one of the following options in order to fully specify the problem:

- *Unidirectionally nearest*: the solution is the index $j \in [1, i-1]$ such that $A[j] > A[i]$ and $i - j$ is minimized. This provides the nearest larger value to the left of the query index, and is equivalent (by symmetry) to the variant in which we ask for the nearest larger value to the right.
- *Bidirectionally nearest*: the solution consists of indices $j_1 \in [1, i-1]$ and $j_2 \in [i+1, n]$ such that $A[j_k] > A[i]$ and $|i - j_k|$ is minimized for $k \in \{1, 2\}$. Thus, in the bidirectional problem, we return both nearest larger values to the left and right.
- *Nondirectionally nearest*: the solution is the index j such that $A[j] > A[i]$ and $|i - j|$ is minimized. As far as we are aware, this formulation has not been considered before.

Furthermore, the data structuring problem has different characteristics depending on whether we consider the elements of A to be distinct (Berkman et al. considered the undirectional variant when all elements in A are distinct).

We consider the problem in the *encoding* model, where once the data structure to answer queries has been created, the array A is deleted. Since it is not possible to reconstruct A from NLV queries on A, the *effective entropy* of NLV queries [2], the log (base 2) of the number of distinguishable NLV configurations, is very low and an NLV encoding of A can be much smaller than A itself. The encoding variant has several applications in space-efficient data structures for string processing, in situations where the values in A are intrinsically uninteresting. Results on encoding NLV problems include (all of the space bounds below are tight to within lower-order terms):

- The bidirectional NLV when A contains distinct values boils down essentially to encoding a Cartesian tree, through which route $2n + o(n)$-bit and $O(1)$-time data structures exist [3,4].
- The unidirectional NLV when A contains non-distinct values can be encoded in $2n + o(n)$ bits and queries answered in $O(1)$ time [5,6]. For the unidirectional NLV the bound is tight even when all values are distinct: we can perturb any instance of the unidirectional problem with non-distinct values in such a way as to preserve the solutions to all queries.[1]
- The bidirectional NLV for the case where elements in A need not be distinct was first studied by Fischer [7]. His data structure occupies $\lg(3 + 2\sqrt{2})n + o(n) \approx 2.544n + o(n)$ bits of space,[2] and supports queries in $O(1)$ time.

In this paper, we consider the nondirectionally nearest larger value (NNLV) problem, in the case that all elements in A are distinct. The above results already hint at the combinatorial complexity of NLV problems. However, the NNLV problem appears to be even richer, and the space bound appears not only to depend upon whether the elements of A are all distinct or not, but also upon the specific tiebreaking rule to use if there are two equidistant nearest values to the query index i.

For instance, given a location i where there is a tie, we might always select the larger value to the right of location i to be its nearest larger value. We call this *rule I*. We give an illustration in the middle panel of Fig. 1 (on page 4). Alternative tie breaking rules might be: to select the smallest of the two larger values (*rule II*); to select the larger of the two larger values (*rule III*); or to select an arbitrary larger value (*rule IV*). Interestingly, it turns out that the tie breaking rule is important for the space bound. That is, if we count the number of distinguishable configurations of the NNLV problem for the various tie breaking rules, then we get significantly different answers. We counted the number of distinguishable configurations subject to rules I–III, for problem instances of size $n \in [1, 12]$, and got the sequences presented in Table 1.

Unfortunately, none of the above sequences appears in the Online Encyclopedia of Integer Sequences.[3] Consider the sequence generated by some arbitrary tie breaking rule. If z_i is the i-th term in this sequence, then $\lim_{n\to\infty} \lg(z_n)/n$ is the constant factor in the asymptotic space bound required to store all the answers to the NNLV problem subject to that tiebreaking rule.

[1] More details: given any array with non-distinct elements for the unidirectional problem, we first reduce the values of the elements to their ranks (allowing ties). We then tweak the values so that the rightmost of each duplicated value x is $x + \varepsilon$ for some $\varepsilon \in (0, 1)$. We then reduce ε by some positive amount such that it is remains positive, and repeat this step until all elements are distinct.

[2] We use $\lg x$ to denote $\log_2 x$.

[3] https://oeis.org/.