Algorithmica (2018) 80:1983–1985
https://doi.org/10.1007/s00453-017-0381-6

Guest Editorial: Special Issue on Compact Data Structures

Travis Gagie[1] · **Gonzalo Navarro**[2]

Received: 19 September 2017 / Accepted: 6 October 2017 / Published online: 10 October 2017

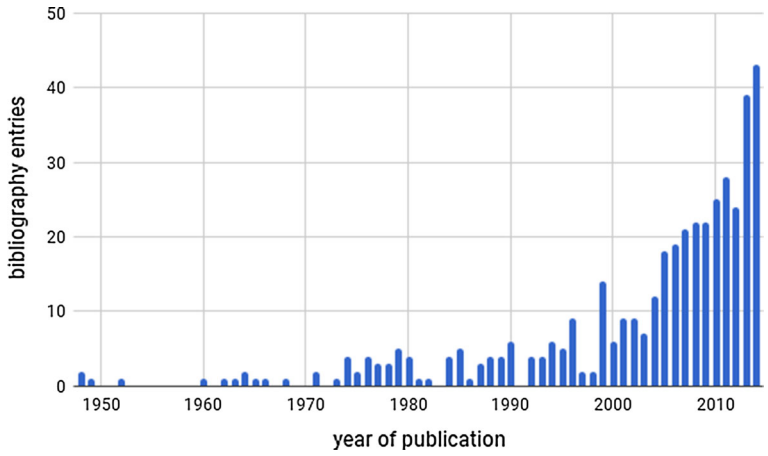
Humanity's thirst for data has driven amazing advances in storage technologies and we can now record more in a year than we did in our whole history up to the start of this millennium. We still want to store even more, however, with faster access to it, while spending less, and without waiting for next year's hardware. In fact, in some fields—perhaps most notably e-commerce but also astronomy, climatology, molecular biology and particle physics, among many others—advances in data collection are even more amazing than those in storage, so if we simply wait a year then the challenge facing us will just grow. This has pushed computer scientists to start making software more space-efficient and, in particular, to devise compact versions of data structures. The bibliography of the second editor's recent textbook, *Compact Data Structures: A Practical Approach*, is evidence of how this field has flourished: the graph below shows the number of books and papers cited that were published in each year from 1948 to 2014.

✉ Travis Gagie
travis.gagie@gmail.com

Gonzalo Navarro
gnavarro@dcc.uchile.cl

1 EIT, Diego Portales University, Av. Ejercito 441, 8370191 Santiago, Chile

2 DCC, University of Chile, Av. Beauchef 851, 8370456 Santiago, Chile



This is the second special journal issue on compact data structures. The first, volume 43 of the *Journal of Discrete Algorithms*, was published in March 2017 and contained six selected papers from special sessions at the Data Compression Conference in 2014 and 2016. The current issue contains five other papers from those sessions (of eight originally selected and invited), the full versions of which are generally longer and somewhat more complex (and have thus taken somewhat longer to review and revise even with the attention of two editors). We expect future special sessions will lead to more special issues, continuing to represent this exciting field.

Three of the papers in this issue are concerned with Lempel–Ziv parsing, one of the most versatile tools in data compression, and the other two give compact representations of trees and graphs with bounded clique-width, that support fast queries.

In the first paper, Alberto Policriti and Nicola Prezza show how to compute the LZ77 parse of a string from run-length encoding of the string when its characters have been permuted with the Burrows–Wheeler Transform (BWT), another versatile compression tool. They give two algorithms that each use $O(n \log r)$ time and $O(r \log n)$ bits of space, where n is the length of the uncompressed string and r is the number of runs in the BWT. Thus, they show how to turn one compressed representation of the string into another, good for different tasks, without decompressing the string (which could take orders of magnitude more space for repetitive strings).

In the second paper, Arz and Fischer revisit the classic problem of storing a dictionary of strings such that, first, given an index, we can quickly return the string with that index in the dictionary and, second, given a target string, we can quickly return its index if it is contained in the dictionary. They show how we can store the dictionary compressed with LZ78 and still support the queries in optimal time, achieving particularly good practical results when the strings in the dictionary contain many long repeated substrings.

In the third paper, Fischer, I, Köppl and Sadakane show how the LZ77 and LZ78 parses of a string can be computed in linear time using either $O(n \log \sigma)$ or $(1 + \epsilon)n \lg n + O(n)$ bits of space, excluding the space for the input but including that for the output, where n is again the length of the uncompressed string and σ is the size

of the alphabet. Their techniques, involving space-efficient construction of succinct suffix trees, also seem likely to be of independent interest.

Lohrey, Maneth and Reh take us beyond the realm of strings and show in the fourth paper how we can represent a tree by a context-free grammar (which can be exponentially smaller than the tree itself) and still support navigation queries—moving to a node's parent or to a specific child, or returning its label—and subtree-equality queries in constant time.

In the fifth and final paper, Kamali moves further beyond strings and shows how we can represent a graph with n vertices and clique-width k in $O(kn)$ bits while still supporting navigation queries quickly: determining nodes' adjacency in constant time, listing their neighbours in constant time per neighbour, and finding their degrees in $O(k \log^* n)$ time.

We thank the organizers of the Data Compression Conference, the participants in the special sessions, the authors of these selected papers, the reviewers, and Ming-Yang Kao and the staff of *Algorithmica*. Without all their efforts, this issue could never have been realized. It has been again an honour and pleasure for us to work with them toward the consolidation and advancement of the field of compact data structures and, thus, toward adapting software to this data-intensive era.