

# Deletions in Random Binary Search Trees: A Story of Errors

Wolfgang Panny

Department of Information Systems and Operations  
University of Business Administration and Economics

Augasse 2-6

A-1090 Vienna

wolfgang.panny@wu-wien.ac.at

Binary search trees (BSTs) are among the most prominent and commonly used data structures for symbol table algorithms. The usual search- and insertion algorithms for BSTs are quite efficient for this purpose. BSTs are especially suitable for applications where (apart from accessing and dynamically inserting the symbols into the table) it is also required to linearly process them according to their sort sequence, e.g., to print a sorted list of the symbols, say. The first publications on BSTs appeared in the early sixties and are due to *P. F. Windley (1960)*, *A. D. Booth* and *A. J. T. Colin (1960)*, and *T. N. Hibbard (1962)*. Each of these papers comprises among other things a description of binary tree insertion and the expected number of key comparisons thereby incurred.

*Hibbard* first showed how to realize deletions from a BST in a reasonable way, thereby considerably extending the application range of BSTs. The BST structure is not restricted to access the nodes ‘by key’. They alternatively can be retrieved ‘by rank’, which only requires straight-forward modifications. In this way BSTs can be extended to a versatile and efficient data organisation for the representation and manipulation of linear lists. Other refinements aim at protecting against the  $O(n)$  worst case behavior of the original structure. This is achieved by imposing additional constraints on the shape of the trees to insure an access time of  $O(\log n)$  even in the worst case. The most prominent species of such balanced search trees is due to *G. M. Adelson-Velsky* and *E. M. Landis (1962)*, who devised their scheme as early as 1962.

In this talk only the original BST structure with access functions *search*,

*insert* and *delete* will be considered. The usual (and reasonable) assumption for the average case analysis of binary search trees are *random insertions*. In a BST built by  $n$  random insertions the expected number of key comparisons necessary to access a node is  $2 \ln n + O(1)$ , which is a well-known result already contained in the first papers on BSTs. However, if random insertions are intermixed with *random deletions* the analysis of the resulting BST seems to become much more intricate and involved. At least this is the impression one gets from the publications on the subject since 1962, and it is quite appropriate to speak of a story of errors in this context. In this contribution we shall take a closer look at this story.