Fast and powerful hashing using tabulation

Thorup, Mikkel

Published in:
36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)

DOI:
10.4230/LIPIcs.FSTTCS.2016.1

Publication date:
2016

Document version
Publisher's PDF, also known as Version of record

Document license:
CC BY

Citation for published version (APA):
Fast and Powerful Hashing Using Tabulation

Mikkel Thorup

University of Copenhagen, Dept. of Computer Science, Copenhagen, Denmark
mikkel2thorup@gmail.com

Abstract

Randomized algorithms are often enjoyed for their simplicity, but the hash functions employed to yield the desired probabilistic guarantees are often too complicated to be practical. Here we survey recent results on how simple hashing schemes based on tabulation provide unexpectedly strong guarantees.

Simple tabulation hashing dates back to Zobrist [1970]. Keys are viewed as consisting of \( c \) characters and we have precomputed character tables \( h_1, \ldots, h_q \) mapping characters to random hash values. A key \( x = (x_1, \ldots, x_c) \) is hashed to \( h_1[x_1] \oplus h_2[x_2] \ldots \oplus h_c[x_c] \). This scheme is very fast with character tables in cache. While simple tabulation is not even 4-independent, it does provide many of the guarantees that are normally obtained via higher independence, e.g., linear probing and Cuckoo hashing.

Next we consider twisted tabulation where one character is 'twisted' with some simple operations. The resulting hash function has powerful distributional properties: Chernoff-Hoeffding type tail bounds and a very small bias for min-wise hashing.

Finally, we consider double tabulation where we compose two simple tabulation functions, applying one to the output of the other, and show that this yields very high independence in the classic framework of Carter and Wegman [1977]. In fact, w.h.p., for a given set of size proportional to that of the space consumed, double tabulation gives fully-random hashing.

While these tabulation schemes are all easy to implement and use, their analysis is not.

This invited talk surveys results from the papers in the reference list. The reader is referred to [8] for more details.


Keywords and phrases Hashing, Randomized Algorithms

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2016.1

Category Invited Talk

References


* Research is partly supported by Advanced Grant DFF-0602-02499B from the Danish Council for Independent Research.


