HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI
MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA
MATEMATISK-NATURVETENSKAPLIGA FAKULTETEN
FACULTY OF SCIENCE

String Sorting in Python – Comparison of Several Algorithms

Onni Koskinen, Arturs Polis, and Lari Rasku

TESTING DATA

Dataset	Number of strings	alphabet size	Sum of LCP array
dna.100MB	618	15	4501
dna.200MB	1114	15	8948
proteins.100MB	359505	24	18853436
proteins.200MB	709116	24	50076184
urls.100MB	3284368	114	94113004
urls.200MB	6576059	114	191545831
words.100MB	18502734	211	83643408
words.200MB	37003241	220	168115390

Table 1: Data set used for comparing the algorithms

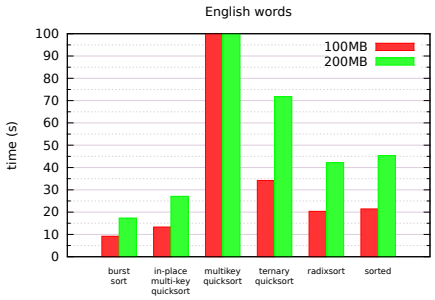
PERFORMANCE GRAPHS

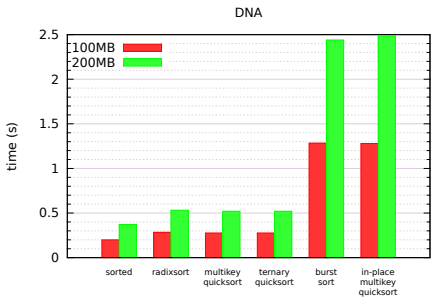
The graphs below show the time and space requirements of several algorithms on two texts. The algorithms are divided into three groups:

New algorithms based on reference point ranks, repetition shortcuts and wavelet trees

Improved implementations of wavelet trees and algorithms from [?]

Prior algorithms from [?, ?]





TEST RESULTS

	sorted (Python builtin)	MSD Radix sort	Multikey quicksort	Ternary quicksort	Burst sort	In-place mulitkey quicksort
dna.100MB	0.2	0.284	0.276	0.276	1.284	1.28
dna.200MB	0.372	0.532	0.52	0.52	2.44	2.484
proteins.100MB	0.768	7.024	7.2	1.908	8.705	4.252
proteins.200MB	1.532	20.921	23.301	3.272	24.67	10.793
urls.100MB	5.072	10.893	25.062	8.585	8.185	5.348
urls.200MB	10.601	21.641	64.836	16.921	16.245	11.697
words.100MB	21.449	20.357	125.384	34.182	9.193	13.313
words.200MB	45.311	42.147	367.687	71.788	17.361	27.09

Table 2: Running times for each algorithm with different data sources