Algorytmy Optymalizacji Dyskretnej 2022/2023

Piotr Kocia

28th May 2023

Contents

1 Introduction

The main focus of this report is single source shortest paths. We have implemented 3 variants of the Dijkstra algorithm and compare them using the data from the 9th DIMACS Implementation Challenge.

2 Algorithms

The three algorithms implemented are Dijkstra, Dial and Radix.

2.1 Dijkstra

Dijkstra's algorithm is the classic best-first algorithm for finding single source shortest paths. We use a priority queue to achieve a monotonic sequence of vertices being visited. The complexity of this version is $O(V \log V + E)$ where V is the number of vertices and E is the number of edges.

2.2 Dial

Dial's algorithm is a modification of Dijkstra that uses a bucket queue instead of a priority queue to improve insertion and extraction time. The complexity is $O(VW)$ where V is the number of vertices and W is the maximum edge weight. In the implementation we create buckets on demand, however, for large VW there might be insufficent memory to accomodate all buckets, which happened several times during our tests.

2.3 Radix

Similarly to Dial, Radix uses buckets, however, the idea is to label the buckets with exponentially increasing value ranges, as opposed to Dial in which case we used ranges of 1. We use powers of 2 as the ranges and label our buckets in a way that makes the range lengths $1, 1, 2, 4, 8, ...$ We use this property to redistribute the elements from the larger to the smaller buckets. The complexity if $O(V \log V + E)$.

3 Results

From the charts its clear that the Dial agorithm does not perform well. In fact, it works most of the time so terribly, that we had been unable to finish gathering data before it was killed. Hence we gave up on using it. Overall, radix and binary heap have its strength and weaknesses that show throughout the tests.

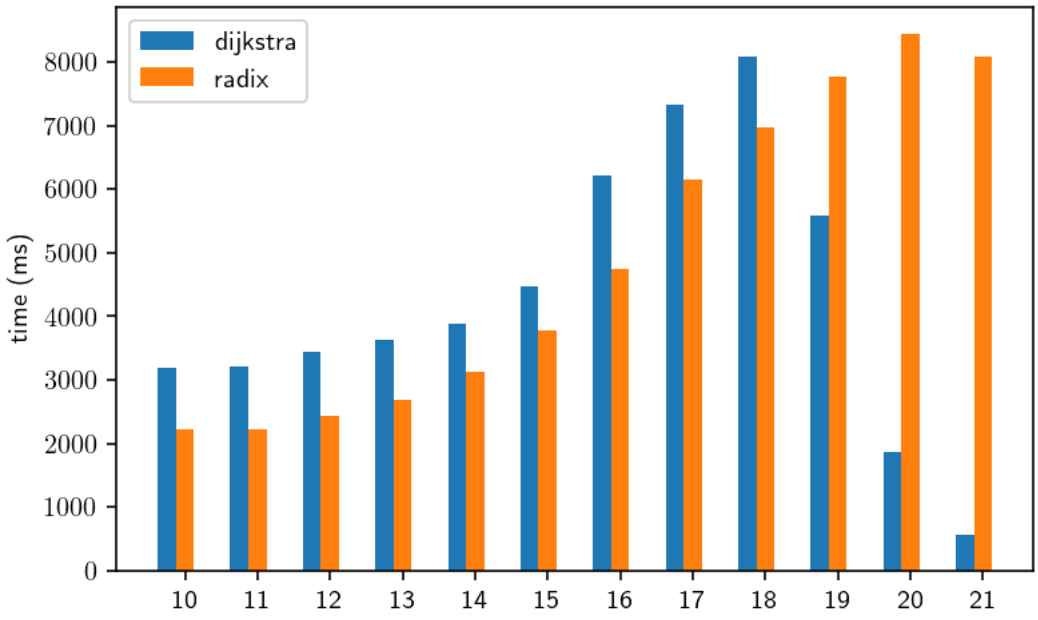

Figure 1: Time to solve all queries in the "one to all" problem for the Long-n family.

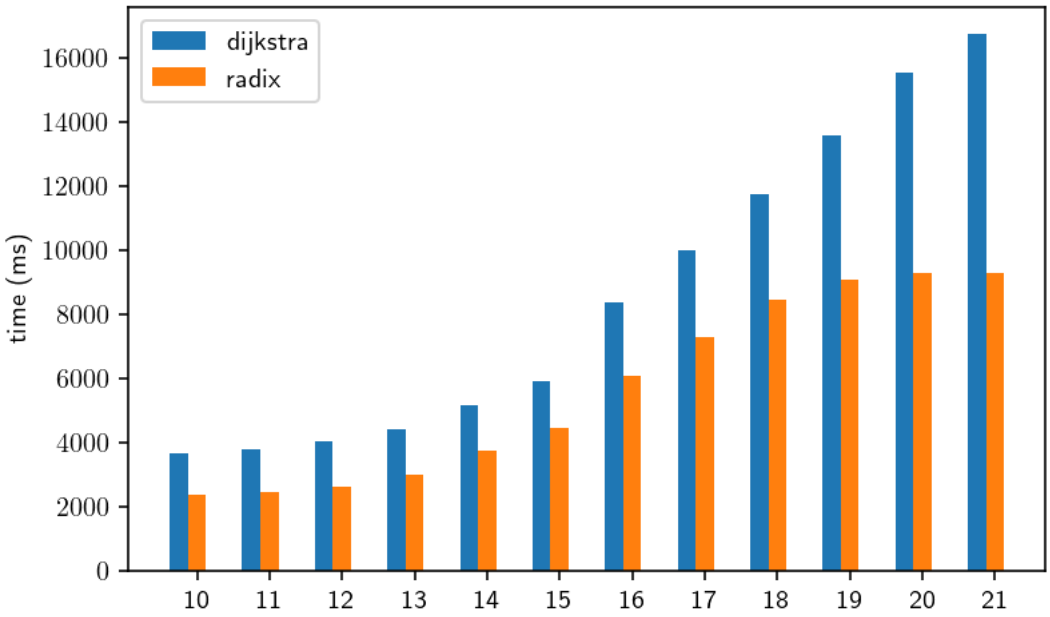

Figure 2: Time to solve all queries in the "one to all" problem for the Square-n family.

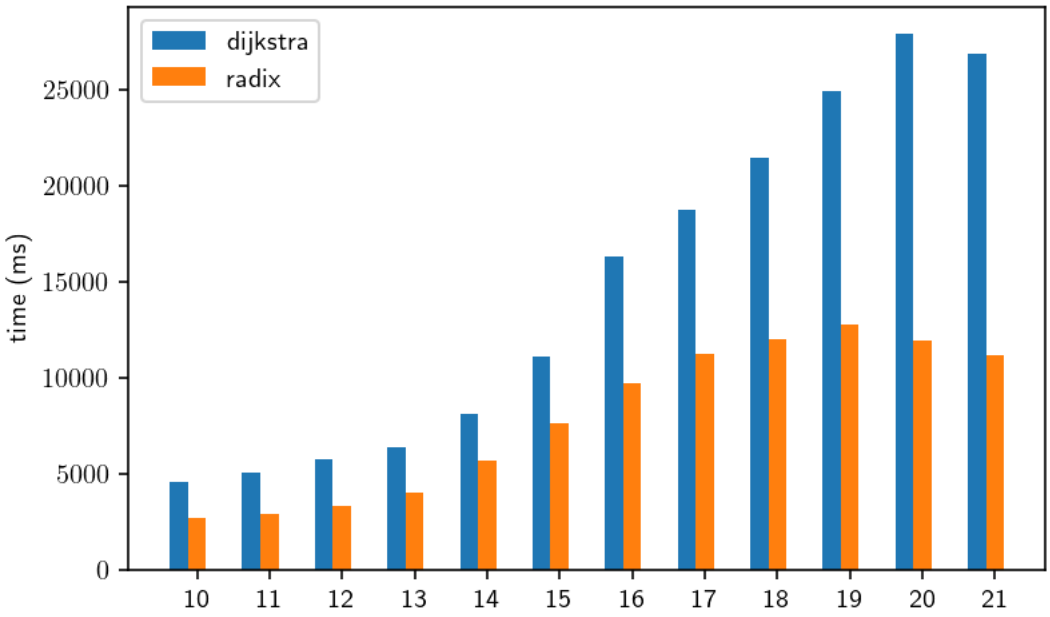

Figure 3: Time to solve all queries in the "one to all" problem for the Random-n family.

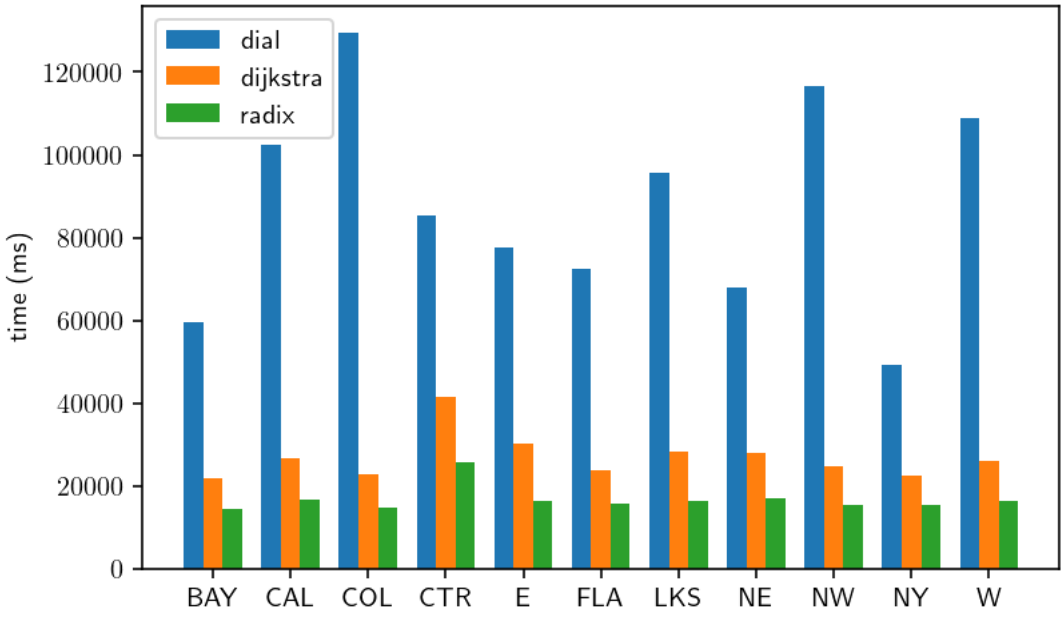

Figure 4: Time to solve all queries in the "one to all" problem for the USA-d family.