Parallel Odd-Even Sort

Lorenzo Beretta (`lorenzo2beretta@gmail.com`)

6th July 2020

1 Introduction

In this report we implemented two parallel versions of the Odd-Even Sort algorithm. The algorithm is pretty simple and works as follows: we proceed repeating identical iterations, each of which consists of two scans over the vector to be sorted; the first scan compare elements in even positions with their successors and, if out of order, swaps them, the same happens for odd-positioned elements in the subsequent scan. For the rest of this report we assume to sort a vector v of length n; it can be proven (https://en.wikipedia.org/wiki/Odd%E2%80%93even$_sort$) that n such identical iterations are sufficient to sort the vector, leading to a sequential complexity of $\mathcal{O}(n^2)$, sub-optimal with respect to the well known $\mathcal{O}(n\log(n))$ alternatives. However this algorithm presents a strong data independence (i.e. its data flow graph is moderately interconnected), therefore we can aim at achieving interesting speedups providing a parallel version.

2 Design Choices

3 Pthread Version

4 FastFlow Version

5 Experiments

6 Conclusions