



UNIVERSITÀ DEGLI STUDI DI MILANO

GPU COMPUTING PROJECT

Bitonic Sort using Numba on Nvidia GPUs

**Author:**

MANUELE LUCCHI

08659A

ACADEMIC YEAR 2022/2023

# 1 Abstract

This documents describe the performances of an existing algorithm, the Bitonic Sort, implemented using Python and Numba, comparing it with a serial approach.

## Contents

<b>1</b>	<b>Abstract</b>	<b>1</b>
<b>2</b>	<b>Introduction</b>	<b>1</b>
<b>3</b>	<b>Algorithm</b>	<b>1</b>
3.1	Step 1: Bitonic Sort . . . . .	2
3.2	Step 2: Bitonic Merge . . . . .	2
<b>4</b>	<b>Implementation</b>	<b>3</b>
4.1	Sequences of length not power of 2 . . . . .	3
<b>5</b>	<b>Benchmark and Profiling</b>	<b>3</b>
<b>6</b>	<b>Conclusion</b>	<b>3</b>

## 2 Introduction

The Bitonic Mergesort [1] is an algorithm created in ANNO by Ken Batcher [2]. The elements to choose for the comparison are independent from the value (it's not data-dependant) therefore is well suited for parallel processing.

## 3 Algorithm

The algorithm is based on the concept of Bitonic Sequence [1], a sequence with  $x_0 \leq \dots \leq x_k \geq x_{k+1} \geq \dots \geq x_{n-1}$  for some  $k$ ,  $0 \leq k \leq n$ , meaning that there are two subsequences sorted in opposite directions.

By using a sorting network, we can create a Bitonic Sequence from any sequence and then merging them to obtain the final sorted sequence, so the algorithm has two phases.

This algorithm has an asymptotic complexity of  $O(n \log(n)^2)$ , the same of the odd-even mergesort [3] and shellsort [4]

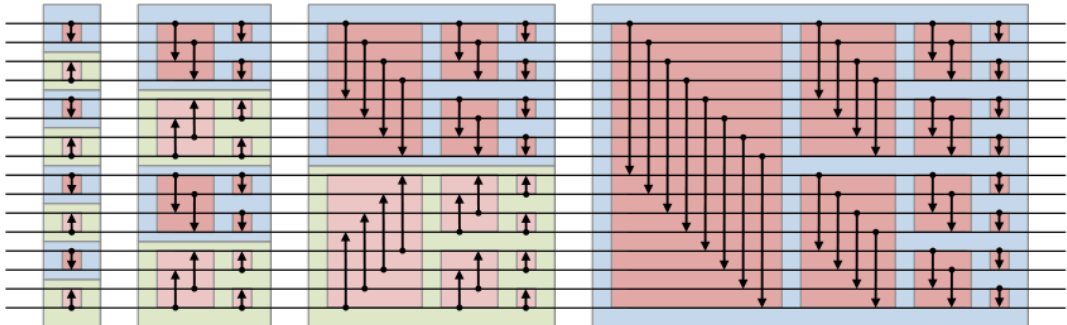
### 3.1 Step 1: Bitonic Sort

To create a Bitonic Sequence we need to build a sorting network. This step has  $\log_2 n$  phases (with the last one being the second step), each one with  $i$  stages, where  $i$  is the current phase starting from 1. By sorting each pair of elements in the sequence in different directions pairwise (using the so called "comparers"), we obtain a sequence full of bitonic subsequences. We can then at each phase double the size of these subsequences and half their number.

As we can see in picture one, at each stage of a phase the elements in the comparers get closer and in each phase a bitonic subsequence is sorted. This structure is known as butterfly network [5]

### 3.2 Step 2: Bitonic Merge

The last step is a variation of the first one, where we only have a single bitonic sequence and sort the two subsequences with the comparers oriented in the same direction, resulting in the sorted sequence.



**Figure 1:** The structure of a bitonic sorting network

## 4 Implementation

### 4.1 Sequences of length not power of 2

## 5 Benchmark and Profiling

Size	CPU Recursive	CPU Iterative	GPU
10	10 ms	10 ms	10 ms
10	10 ms	10 ms	10 ms

Size	Cuda C++	Python Numba
10	10 ms	10 ms
10	10 ms	10 ms

## 6 Conclusion

## References

1. Einstein, A. Zur Elektrodynamik bewegter Körper. (German) [On the electrodynamics of moving bodies]. *Annalen der Physik* **322**, 891–921 (1905).
2. Knuth, D. *Knuth: Computers and Typesetting* <http://www-cs-faculty.stanford.edu/~uno/abcde.html>. (accessed: 01.09.2016).
3. Knuth, D. *Knuth: Computers and Typesetting* <http://www-cs-faculty.stanford.edu/~uno/abcde.html>. (accessed: 01.09.2016).
4. Knuth, D. *Knuth: Computers and Typesetting* <http://www-cs-faculty.stanford.edu/~uno/abcde.html>. (accessed: 01.09.2016).
5. Knuth, D. *Knuth: Computers and Typesetting* <http://www-cs-faculty.stanford.edu/~uno/abcde.html>. (accessed: 01.09.2016).