Merge Binary Insertion Sort Library

Merge Binary Insertion Sort is an hybrid sorting algorithm which combines Binary Insertion Sort and Merge Sort. This algorithm switches from Merge Sort to Binary Insertion Sort for sequences of items whose length is less than a certain threshold value, taking advantage of the latter's better performance over smaller lists.

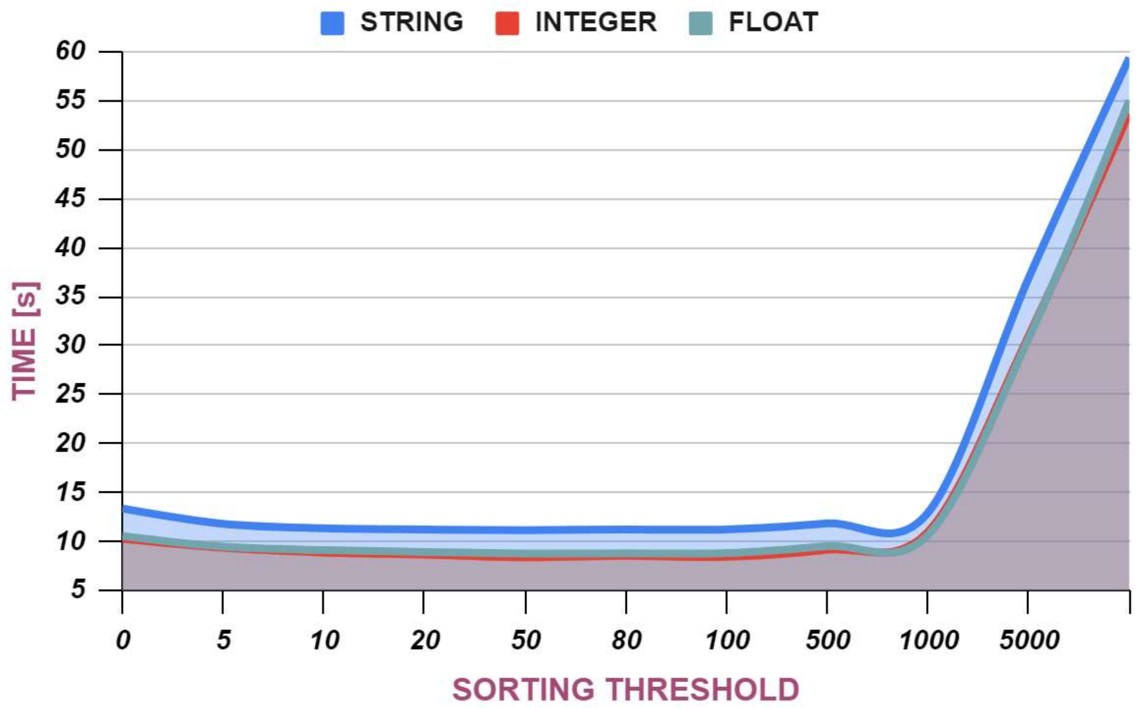
The library provides an implementation of this algorithm in C, and allows the user to manually specify this threshold value to fine-tune the execution of Merge Binary Insertion Sort over a sequence of generic items. The value of threshold can increase or decrease the performance of the algorithm, and it's important to choose the optimal value based on the specific usage scenario.

Below is a performance analysis in terms of execution time for the algorithm operating on an array containing 20 million items. The analysis involves varying both the threshold value and the type of items being compared.

	Sorting Threshold										
TYPE		0	5	10	20	50	80	100	500	1000	5000	10000
	STRING	13,357	11,807	11,347	11,220	11,148	11,230	11,222	11,845	12,929	36,913	59,403
	INTEGER	10,264	9,355	8,872	8,665	8,392	8,529	8,429	9,156	10,973	31,008	53,783
	FLOAT	10,546	9,459	9,113	8,919	8,768	8,785	8,807	9,517	10,666	30,748	55,025

Note that, when the threshold value is 0, the Binary Insertion Sort is not used anywhere during the execution. It can be observed how using threshold values slightly larger than zero improves the performance by one second. Further, optimizing the threshold up to around 50 yields the most favorable results. On the other hand, increasing the threshold value by too much will cause a significant performance loss. It is evident in the observed time increments, particularly notable when comparing the values of 1000 to that of 5000, and the values of 5000 to that of 10000, where a drastic rise in execution time is apparent.

This plot illustrates the execution time of the sorting algorithm in correlation with the threshold value for the three analyzed types.



The functions used for elements comparison are:

- The standard comparison operators for *integers* and *floats*;
- The library function `strcmp` for *strings*.

As we can see, the curves are practically equivalent, with the one for the string datatype slightly shifted up due to the performance overhead of the `strcmp` function.

The provided table outlines the best, worst, and average time durations for the sorting algorithm, along with the corresponding threshold values for both best and worst scenarios. Interestingly, it is noteworthy that for each considered data type, the optimal and suboptimal threshold values align, indicating a consistency across different scenarios.

		BEST		WORST		AVERAGE
		Time [s]	Threshold	Time [s]	Threshold	Time [s]
TYPE	STRING	11,148	50	59,403	10000	22,099
	INTEGER	8,392	50	53,783	10000	18,610
	FLOAT	8,768	50	55,025	10000	18,902

According to this test, the optimal threshold value should be between **50** and **100**, and should not go beyond 1000. To replicate this test, there shall be a compile-defined macro named `ENABLE_PROFILER` set to 1.