Formal Proof of Correctness: OverflowSort Algorithm

Scott Douglass

July 2025

1. Overview

The OverflowSort algorithm is based on a hardware-inspired model where the number of multiplicative doublings of a number required to exceed a fixed threshold is used as a surrogate for its magnitude. We define this count as the **overflow tick**, and show that sorting based on this tick produces a descending order of the input values.

2. Overflow Tick Function

Let $x_i \in N$ be an input value, and $T \in N$ be the overflow threshold (e.g., $T = 2^{16}$ for 16-bit unsigned integers).

We define the **tick function** as:

$$\text{tick}(x_i) = \min \left\{ t \in N \,\middle|\, x_i \cdot 2^t > T \right\}$$

Rewriting:

$$x_i \cdot 2^t > T \Rightarrow 2^t > \frac{T}{x_i} \Rightarrow t > \log_2 \left(\frac{T}{x_i} \right)$$

Thus:

$$\text{tick}(x_i) = \left\lceil \log_2 \left(\frac{T}{x_i} \right) \right\rceil = \lceil \log_2(T) - \log_2(x_i) \rceil$$

3. Monotonicity and Ordering

We claim: if $x_i > x_j$, then $\text{tick}(x_i) < \text{tick}(x_j)$.

Proof:

Since $\log_2(x)$ is strictly increasing:

$$x_i > x_j \Rightarrow \log_2(x_i) > \log_2(x_j) \Rightarrow \text{tick}(x_i) < \text{tick}(x_j)$$

Conclusion: Sorting by $\text{tick}(x_i)$ gives the reverse of numeric magnitude: highest x_i corresponds to lowest tick value.

4. Invertibility

Given $\text{tick}(x_i) = t$, we know:

$$x_i < \frac{T}{2^{t-1}}$$

This implies each tick value encodes a range of possible x_i, forming a coarse logarithmic bin.

5. Time Complexity of Tick Evaluation

Evaluation of $\text{tick}(x_i)$ can be done using:

- Loop-based doubling: $O(\log x)$ worst case
- Bitwise operations (e.g., count leading zeros): $O(1)$ on modern CPUs

6. Overall Sorting Correctness

Let $\{x_1, x_2, \ldots, x_n\}$ be the input array. Let σ be a permutation such that:

$$\text{tick}(x_{\sigma(1)}) < \text{tick}(x_{\sigma(2)}) < \ldots < \text{tick}(x_{\sigma(n)})$$

Then:

$$x_{\sigma(1)} > x_{\sigma(2)} > \ldots > x_{\sigma(n)}$$

Therefore: OverflowSort produces a correct descending sort by using tick as an order proxy.