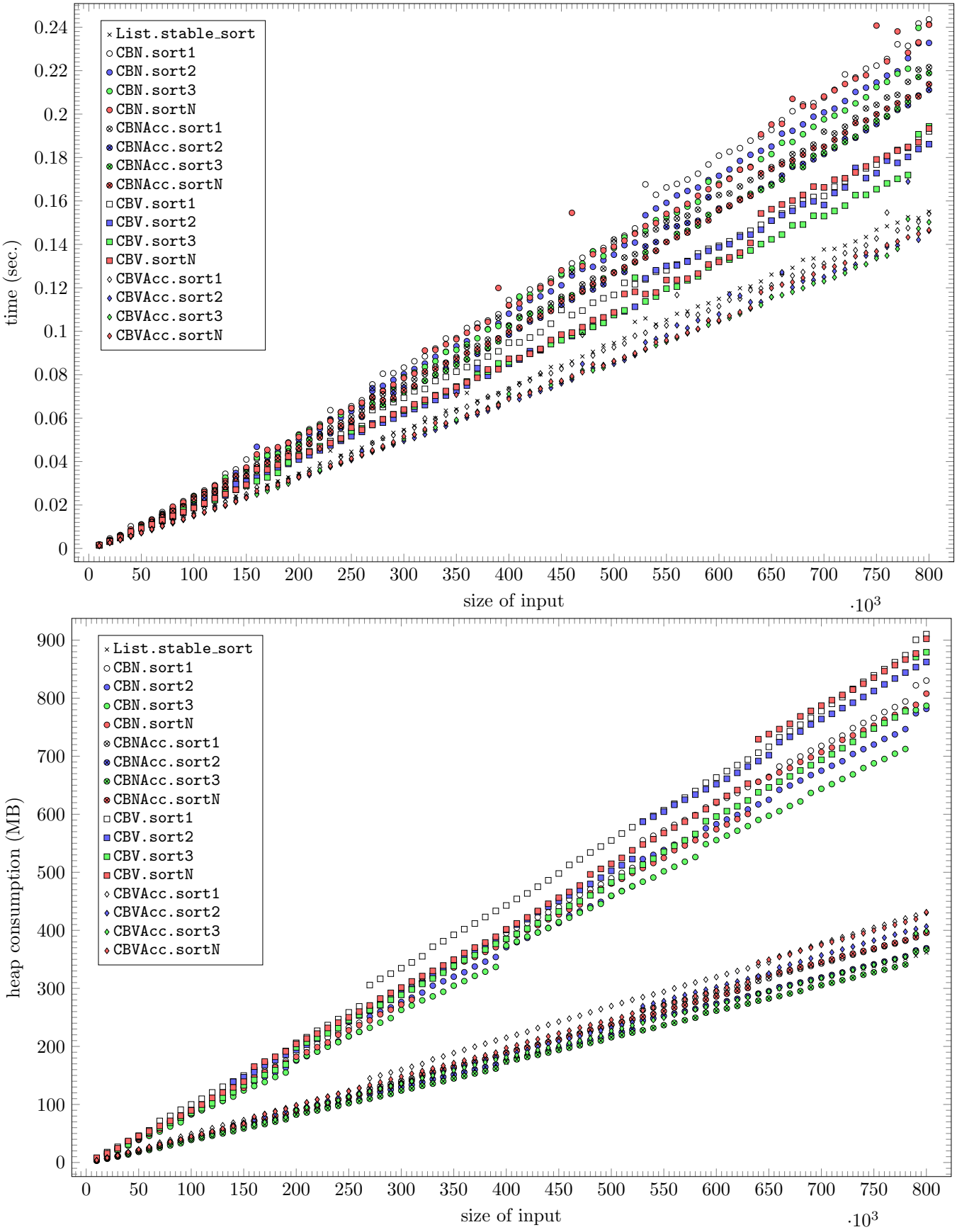

Figure 1: Benchmark result of the sorting functions extracted to OCaml.

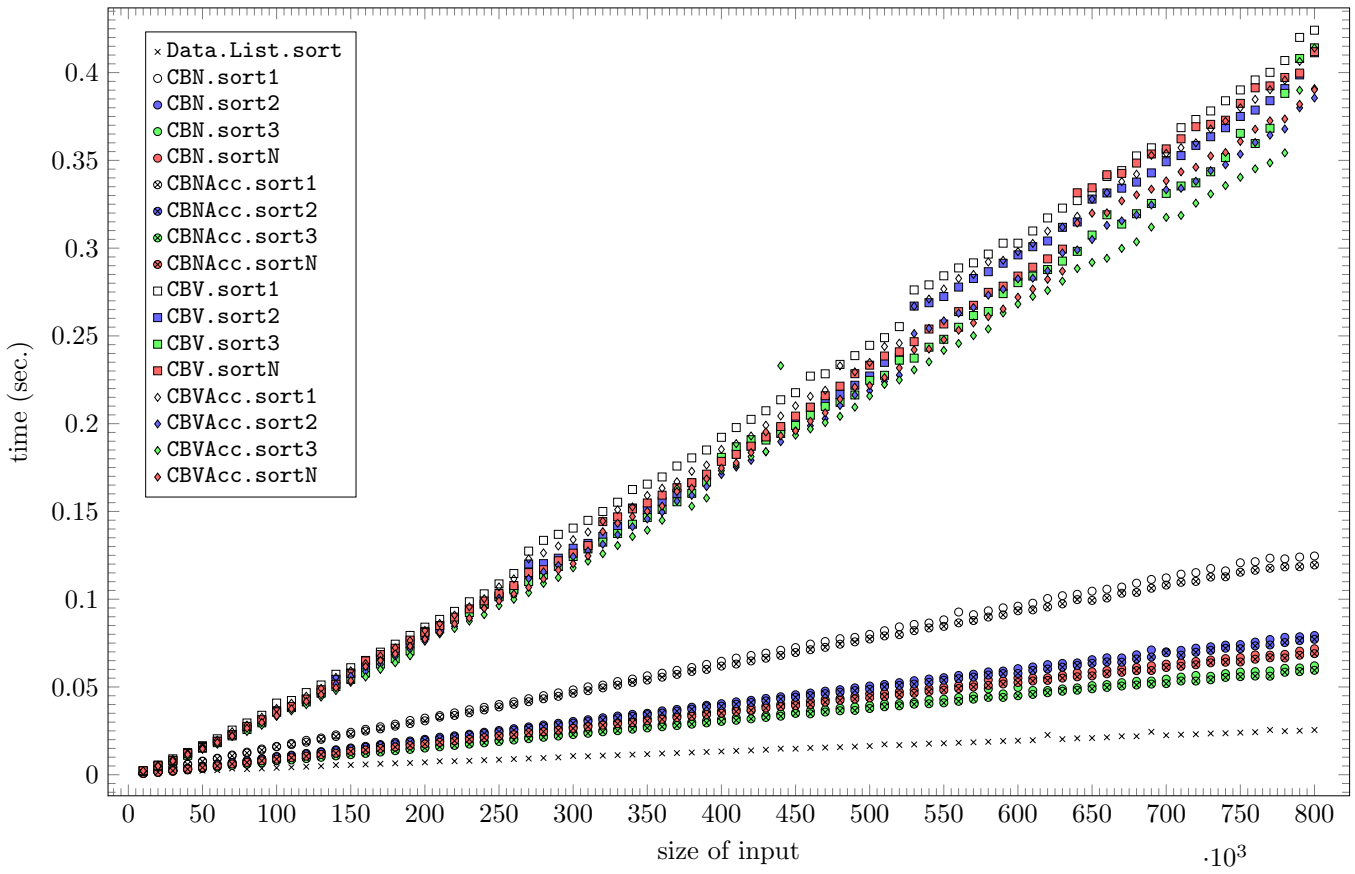

Figure 2: Benchmark result of `sorted (take 1000 (sort xs))` in Haskell, where *sort* is `Data.List.sort` or an extracted sorting function, and `xs` is a random input of type `[Int]`.

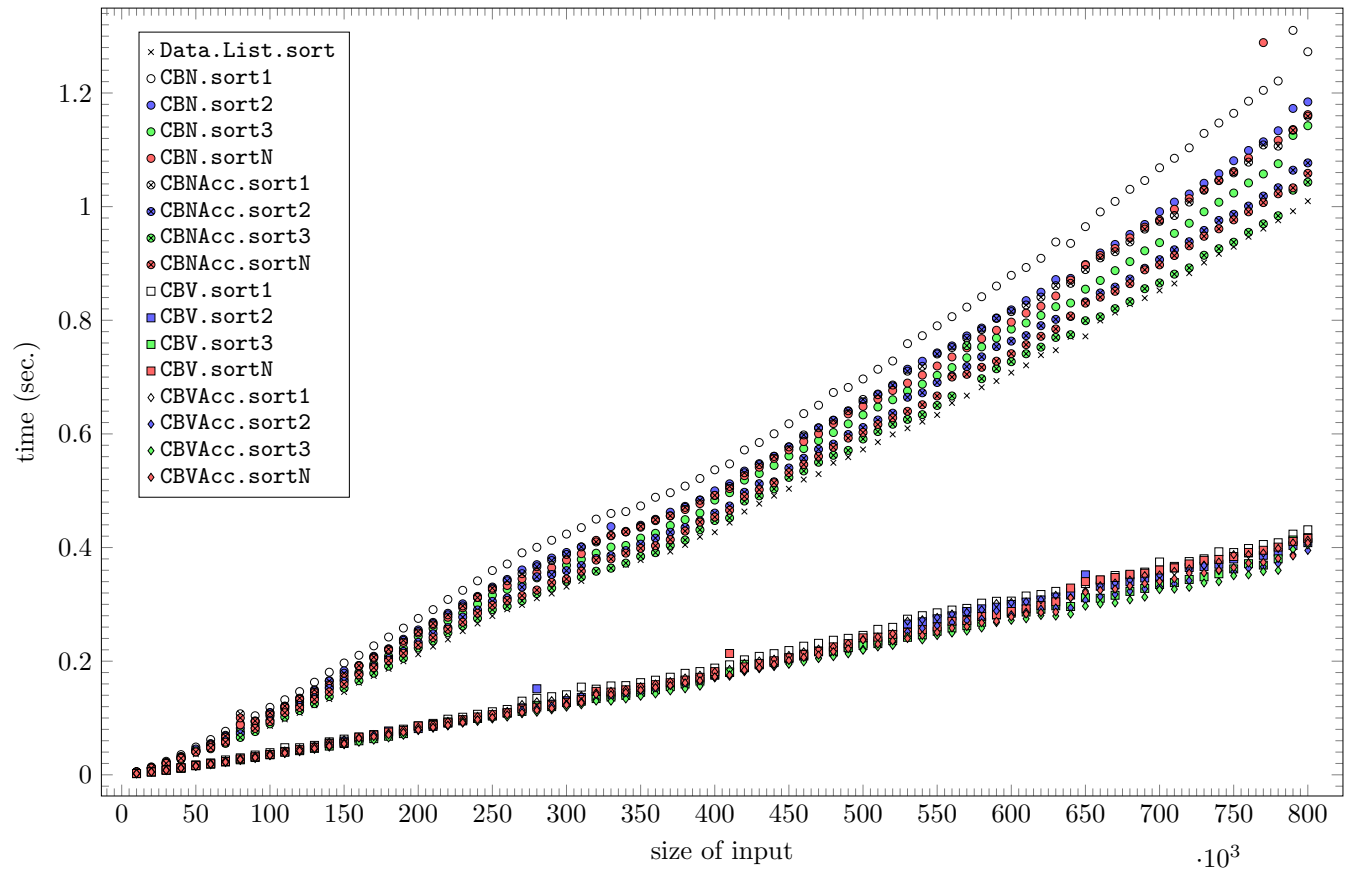

Figure 3: Benchmark result of `sorted (sort xs)` in Haskell, where *sort* is `Data.List.sort` or an extracted sorting function, and `xs` is a random input of type `[Int]`.

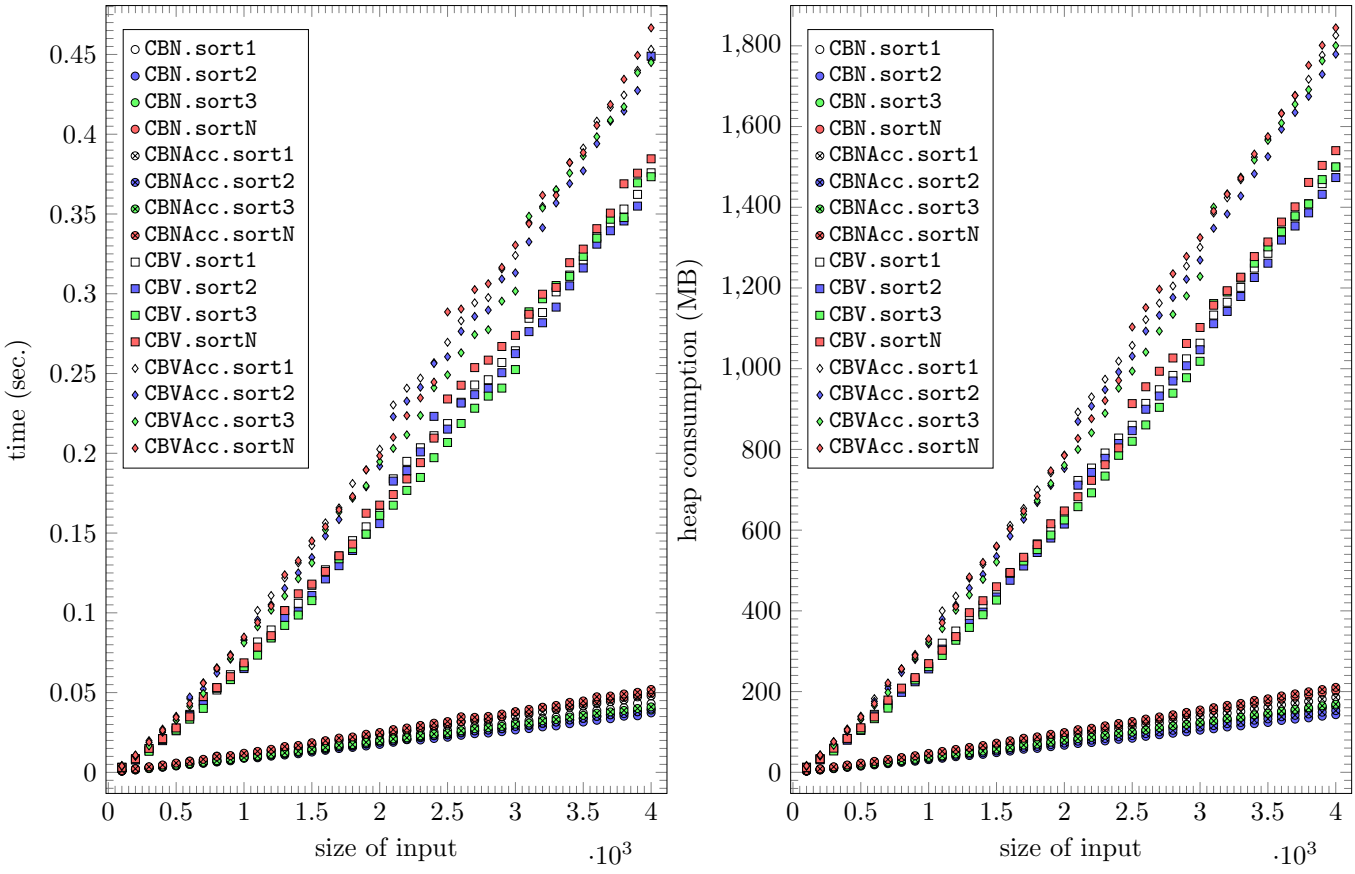

Figure 4: Benchmark result of `sorted N.leb (take 10 (sort N.leb xs))` with `lazy`.

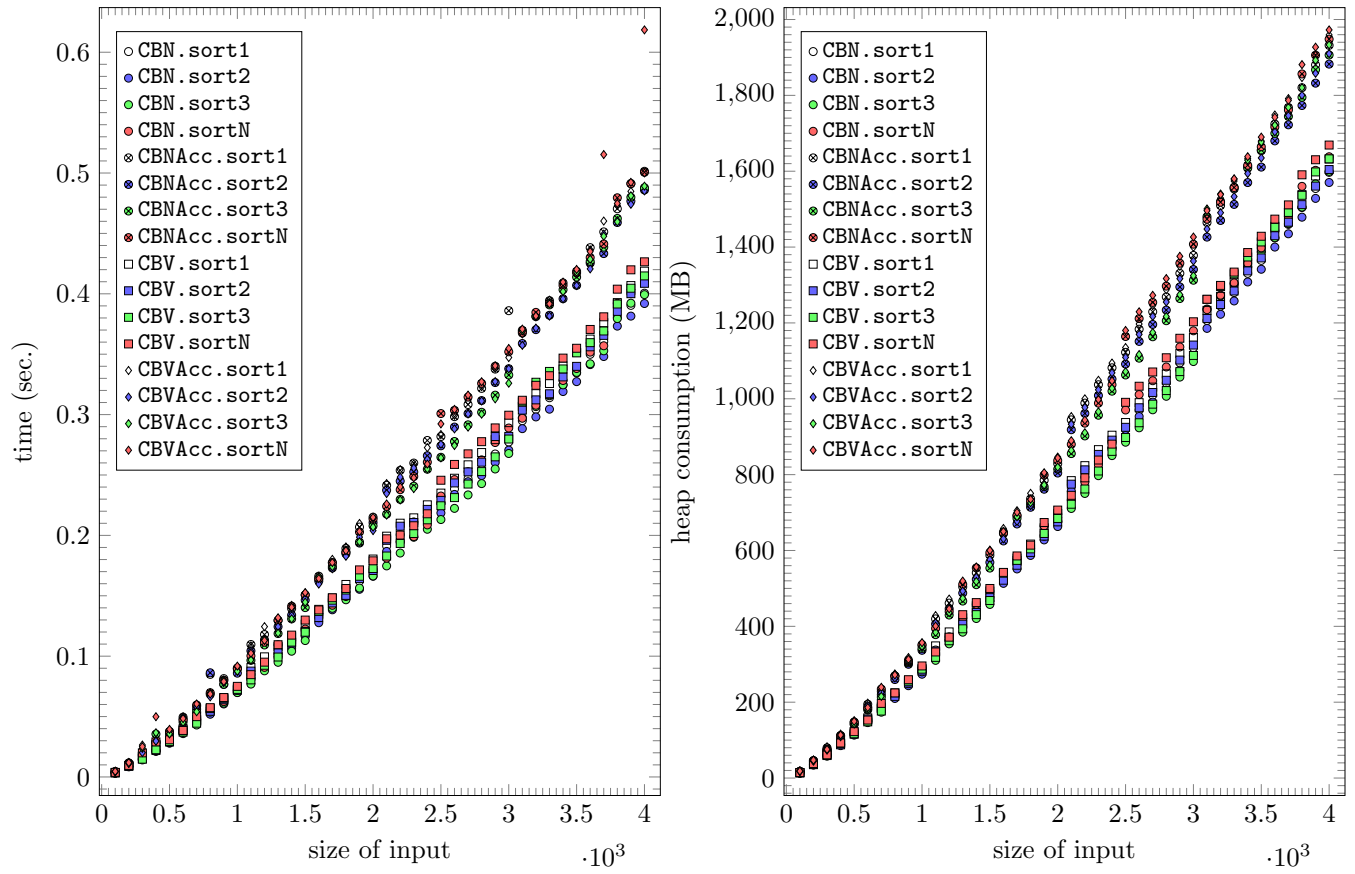

Figure 5: Benchmark result of `sorted N.leb (sort N.leb xs)` with `lazy`.

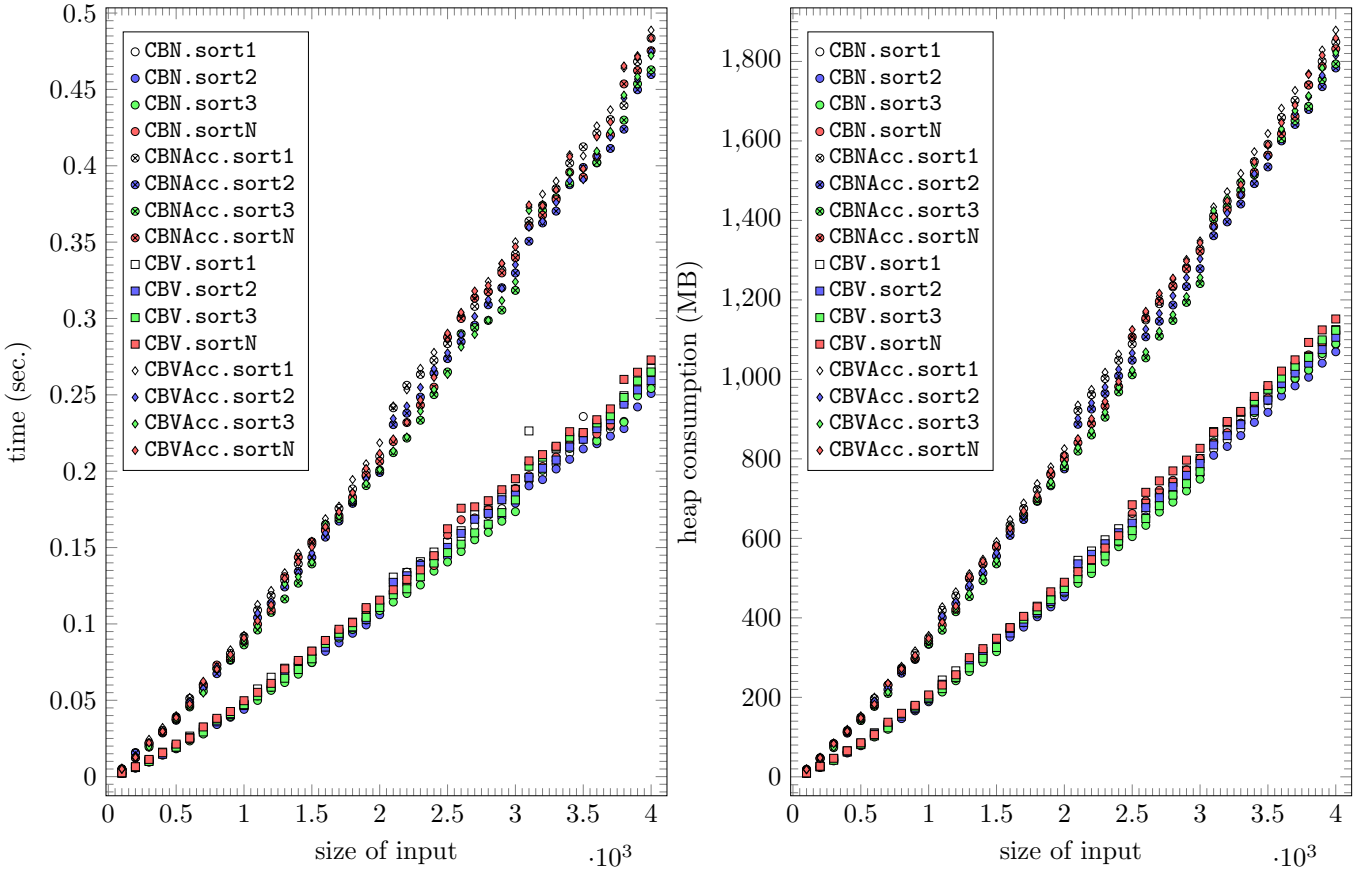

Figure 6: Benchmark result of `sorted N.leb (sort N.leb xs)` with `compute`.

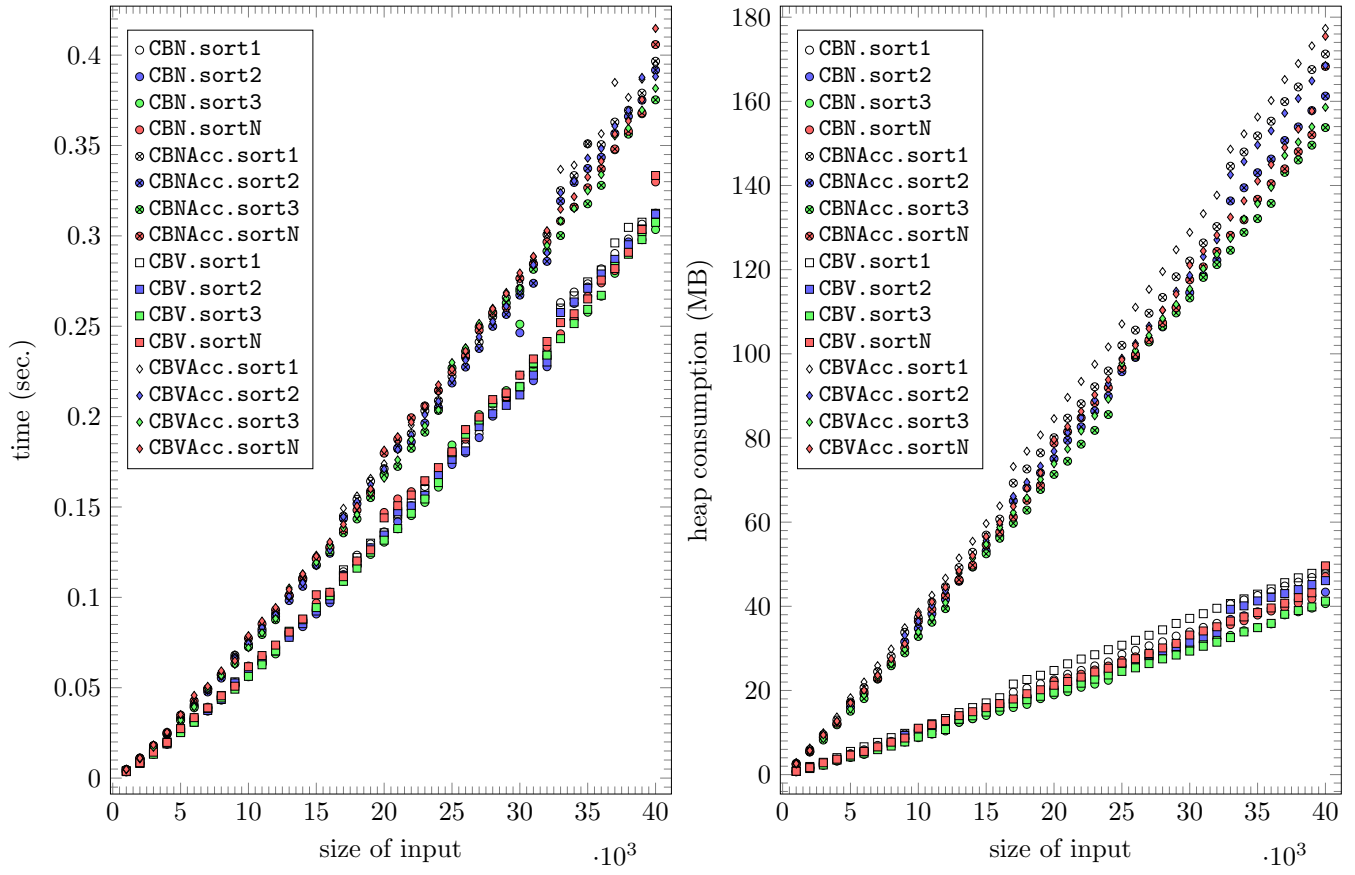

Figure 7: Benchmark result of `sorted N.leb (sort N.leb xs)` with `vm_compute`.

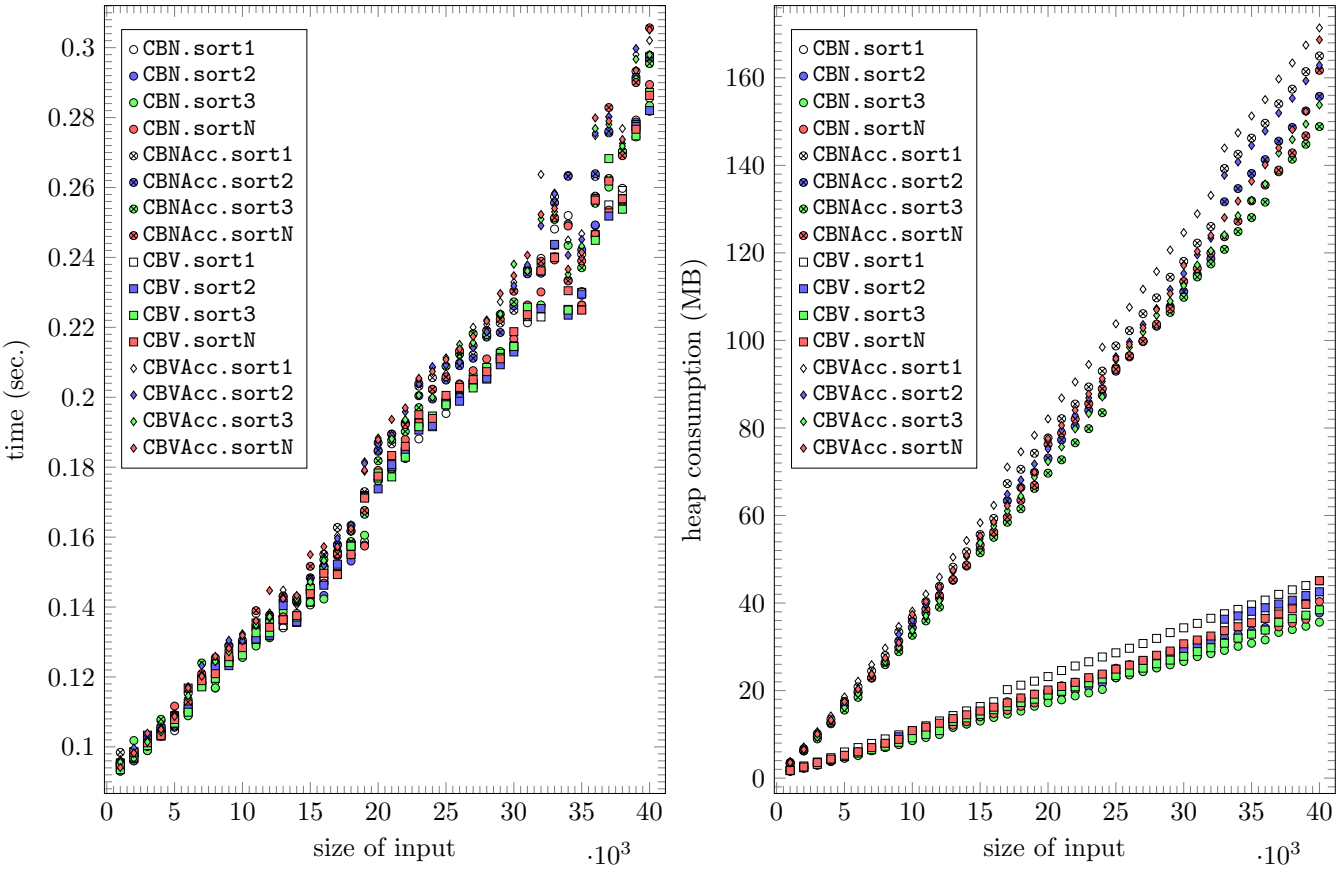

Figure 8: Benchmark result of `sorted N.leb (sort N.leb xs)` with `native_compute`.