$\log n$ - Space, $n^{3/2}$ Time Quantum Sort.

David Ponarovsky

January 29, 2024

It has been proven that any quantum algorithm in the quantum circuits which sorts at time T and storage space S has to satisfy the restriction $TS = \Omega(n^{3/2})$ [Kla03]. In the regime of $S \geq \log^3(n)$, it has been shown that the bound is tight up to logarithmic factors. However, in the regime where S is strictly $\Theta(\log(n))$, not much advancement has been reached beyond $T = \Theta(n^{1\frac{1}{2}} \log n)$. Here, we present a quantum algorithm that sorts with $\log(n)$ storage memory and $n^{3/2}$ time. We achieved this by quantifying the sorting algorithm invented by Stanley P. Y. Fung [Fun21], who coined its name - "ICan'tBelieveItCanSort" - due to the surprise of having such a simple sorting algorithm.

Result: Sorting $A_1, A_2, ..A_n$
for $i \in [n]$ **do**
  for $j \in [n]$ **do**
    if $A_i < A_j$ **then**
      swap $A_i \leftrightarrow A_j$
    end
  end
end

Algorithm 1: "ICan'tBelieveItCanSort" alg.

Result: Sorting $A_1, A_2, ..A_n$
swap $A_1 \leftrightarrow \max A$
for $i \in [n-1]$ **do**
  Find the first k such $A_k > A_i$
  Set $A \leftarrow A_1, A_2..A_{k-1}, A_i, A_k, A_{k+1}, .., A_{i-1}, A_{i+1}.., A_n$
end

Algorithm 2: "ICan'tBelieveItCanSort" alg.

References

[Kla03] Hartmut Klauck. *Quantum Time-Space Tradeoffs for Sorting*. 2003. arXiv: quant-ph/0211174 [quant-ph].

[Fun21] Stanley P. Y. Fung. "Is this the simplest (and most surprising) sorting algorithm ever?" In: *CoRR* abs/2110.01111 (2021). arXiv: 2110.01111. URL: https://arxiv.org/abs/2110.01111.

Result: Sorting $A_1, A_2, ..A_n$

swap $A_1 \leftrightarrow \max A$
for $i \in [n-1]$ **do**
    Set current $\leftarrow$ head.next
    k-pointer $\leftarrow$ Find the first '$k < i$' node such '$A_k > A_i$' using Grover querying the follow
      Ask if (node.color = red and node.value > current.value
        and node.back.value $\leq$ current.value)
    Set head.next $\leftarrow$ head.next.next
    Set head.next.back $\leftarrow$ head
    Set current.next $\leftarrow$ k-pointer
    Set current.back $\leftarrow$ k-pointer.back
    Set current.back.next $\leftarrow$ current
    Set current.color $\leftarrow$ red
end

Algorithm 3: "Quantum ICan'tBelieveItCanSort" alg.