

Referee #1:

The authors of "Large-scale simulation of Shor's quantum factoring algorithm" are long-time, leading contributors in this field, and the clear exposition of their work is a testament to their expertise. With the exception of a few figures and sections where the current manuscript can be improved, we highly recommend this work be accepted for publication.

We are very grateful to the Referee for their enthusiastic review, carefully thought-out questions and comments, and for recommending our manuscript be accepted for publication.

This paper reports a new method of simulating Shor's quantum factoring algorithm on relatively large semiprimes using a supercomputer and many GPUs. By choosing the "iterative" Shor method, they are able to simulate 40 qubits and factor 12-digit numbers efficiently enough to analyze 60000 cases, an impressive achievement. Using this data they are able to study the success probability of the algorithm as well as some of the effects of noise. It is also very nice to see analysis of both the standard Shor post-processing as well as the improved methods from Ekerå.

The paper begins with a very good introduction and overview of the state of the art in quantum factoring. The authors also propose a challenge to actual quantum computers which we find very appealing, namely to factor a semiprime larger than can be done by their simulator. This is an excellent step along the road to true quantum supremacy.

We thank the Referee for appreciating our proposal and for finding the challenge very appealing.

The biggest problem with the paper is in the study of the effects of noise. With their method, they are only able to study initialization and measurement errors. While this is interesting (and measurement errors in particular can be quite large in many quantum computer implementation), most of the work (and therefore the errors) in a real quantum circuit will be in the multiplications. Since the method used to multiply replaces the circuit with a direct permutation it appears that this simulator cannot address errors in the multiplication. The authors should at least acknowledge and discuss this limitation.

We agree with the Referee that, in a real quantum circuit, most of the work will be in the multiplications. For our simulation, we chose to implement it in terms of a direct permutation, which allows the distributed memory scheme and thus enables the study of large-scale factoring scenarios. Unfortunately, it limits our study of errors to initialization errors before and measurement errors after the multiplications. The alternative, namely to implement the multiplications in terms of standard quantum gates, would require additional qubits and simulation time and thus reduce the size of simulatable factoring problems.

Hence, the strength of our method, which enables simulation of the largest instances, at the same time limits the extent to which errors in the corresponding parts of the circuit can be simulated, and we acknowledge that this limitation has not been sufficiently emphasized in our previous manuscript. Therefore, in the revised manuscript, we have added a new section 3.4 "Discussion of Limitations and Future Directions", which elaborates on the limitations and

discusses the alternatives. To pursue the alternative, we believe that a great place to start is <https://doi.org/10.22331/q-2021-04-15-433>, and we thank the Referee for motivating us to mention this path as an exciting future research direction.

The authors are also perhaps too credulous on annealing. The algorithms only scale like L^2 in the number of qubits, but are almost certainly exponential in time and come with no guarantee of finding an answer. We are skeptical that cryptographically interesting numbers will ever be factored on an annealing machine, though perhaps annealers will be able to outmatch this paper's simulator someday. Note that the L^2 scaling is mentioned in both the introduction and conclusion.

We agree with the Referee that the initial manuscript has given a too credulous impression on quantum annealing. The scaling of the CSP/circuitSAT formulation of factoring for quantum annealers including effects from diabatic and thermal transitions has not yet been mapped out, and it could well be that the algorithms are exponential in time. In the revised manuscript, we have therefore removed (i) the speculative statement about cryptographically relevant numbers from the introduction and (ii) the mentioning of exponential scaling from the conclusion.

It would also be very nice if the authors could comment on if their method would work for the algorithm by Ekerå and Hastad in quant-ph/1702.00249, and nicer if they could simulate it and factor even larger numbers.

We thank the Referee very much for suggesting this interesting future research direction. Indeed, an iterative version of the Ekerå-Håstad algorithm could be simulated using an analogous MPI communication scheme (with different permutations depending on a and b). Unfortunately, it would not directly allow the simulation of larger numbers, as the number of qubits that need to be simulated would stay the same—only the number of steps in the semiclassical Fourier transform would decrease, which would save valuable simulation time.

Combined with another optimization, however, it could indeed allow the simulation of larger factoring scenarios. We have discussed these two future research directions under points 2 and 3 in the new section 3.4 “Discussion of Limitations and Future Directions”, which also aligns well with the structural suggestions of the Mathematics journal style template.

Below are some more minor notes and questions that we have:

Page 5 section B: What is meant by “no modular exponentiation needs to be implemented on the quantum part of the iterative Shor algorithm?” and that this is an advantage that iterative Shor has over the conventional algorithm. It seems that the whole circuit in Figure 2 exists to run modular exponentiation. If what is meant is that the circuit has been expanded by the shift-and-multiply algorithm, wouldn't all Shor implementations do this?

We agree with the Referee that the combination of all controlled modular multiplications of the iterative Shor algorithm is, in essence, a modular exponentiation. We have removed the quoted sentences from the revised manuscript. We thank the Referee for catching this point.

What we had in mind when writing this part was that, for known a , the transformations generated by each individual controlled modular multiplication are fully known, so that each operation can be individually optimized to the largest possible extent. Although this is a computationally hard problem in itself, depending on the primitive gate set available, there may be compiled versions of the transformations that may need much less workspace qubits than a general controlled modular multiplication circuit. For instance, in <https://doi.org/10.1126/science.aad9480> and <https://doi.org/10.1103/PhysRevA.100.012305>, the authors use modular multiplication circuits without any additional workspace qubits (albeit only to factor numbers until 35).

Initialization errors: Why not initialize all the bits with errors instead of just the $+$ states?

Initializing the internal qubits to some erroneous version of $|0\dots01\rangle$ is one of the many interesting types of errors that we did not consider. We have added the following motivation to Section 2.7 “Initialization Errors”:

“Our motivation to prioritize the recycled qubit for a study of initialization errors instead of the other “internal” qubits is that this qubit is measured and reinitialized successively in every stage of the iterative Shor algorithm.”

Bottom on page 16: Are the improvements due to further post-processing algorithms due to you (especially the first one has no citation)?

We apologize for this confusion; the “further post-processing algorithms” were meant to refer to Ekerå’s algorithms. We have removed this sentence in the revised manuscript, as it is already stated in the previous paragraph that the probabilities include further post-processing.

Errors are discussed for Ekerå post-processing but not for Shor. Is there a particular reason for this? Similarly, the lucky cases are discussed for Shor but not Ekerå, is there an equivalent for Ekerå?

Regarding the first question, the reason lies in the history of this work: We started the present study because we were skeptical of the usually very small theoretical estimates of the success probabilities for Shor’s algorithm. This is how we encountered the lucky cases. However, from Ekerå we learned that with his additional post-processing, the success probabilities can already be brought close to unity. In other words, for Ekerå’s post-processing, there are no “lucky cases”, as all cases can be led to success by performing appropriate post-processing. This is why we then added the study of errors as a “next level” to Ekerå’s post-processing. For Shor’s post-processing, additional errors would reduce the success+lucky probabilities again towards the original success probabilities, so we did not perform a separate evaluation for the Shor case.

Can the authors make a recommendation of Shor vs Ekerå for actual use? Or should one just use both since they are just post-processing?

For practical applications, the safest option is indeed to try both. However, since Ekerå’s post-processing requires more effort in choosing appropriate parameters and setting up the

algorithms, using Shor's post-processing is certainly simpler in practice. Especially if not just a single run of the quantum part but $O(1000)$ shots are performed (as is often done on current devices which are seen as “samplers”), our investigation shows that the success probability is as large as with Ekerå's post-processing in the presence of errors. Hence, a practical rule could be: if the execution on the quantum computer is very costly, it is probably better to invest in the classical post-processing by Ekerå. If it takes almost no time to get about $O(1000)$ samples (with low enough error-probability), it may be more convenient to simply use the standard procedure.

Page 11: "Furthermore, let r be the denominator.." Where is " k " defined?

We thank the Referee for spotting this omission. We have added the definition in section 2.8.1 and the caption of Figure 1.

Page 27: Why are the semi primes in Table I considered interesting for the iterative algorithm even though p and q have such drastically different number of digits? In fact, it's known that when p or q is not close to $\sqrt{N}$, then the problem can be solved rather quickly with classical approaches.

We apologize for the confusion; the reference was supposed to refer to Table II (the “largest interesting” semiprimes) instead of Table I (the “largest odd” semiprimes). On further reflection, we decided that only Table II is of sufficient relevance in the context of the manuscript, so we have removed the former Table I from the revised manuscript. Indeed, the Referee is right that these cases are relatively easy to factor classically e.g. by trial division.

Finally, here are a few typos to address: Page 6: "one would use much [fewer] GPUs for 7 qubits" Page 11: "the order is known (blue [triangles])"

We thank the Referee for catching these typos. We have corrected them in the revised manuscript.