Applied Physics Research; Vol. 10, No. 4; 2018
ISSN 1916-9639 E-ISSN 1916-9647
Published by Canadian Center of Science and Education

A Classical Mechanism for Shor's Algorithm Implementations

David L. Selke[1]

[1] Carter Logistics, Anderson, Indiana

Correspondence: David Selke, 6548 N 100 W, Alexandria, IN. Tel: 254-423-1250. E-mail: dselke@hotmail.com

Received: May 16, 2018 Accepted: June 16, 2018 Online Published: July 12, 2018

doi:10.5539/apr.v10n4p24 URL: https://doi.org/10.5539/apr.v10n4p24

Abstract

Loops that enforce a correct output and that restart with a changed parameter may emulate a brute force search, even against the design intent. A Python program is presented analogous to Shor's Algorithm but with random number generation replacing the math. It factors integers. Shor's Algorithm devices may operate similarly to the Python program, not in being random, but in being classical.

Keywords: quantum, factoring, loophole, Shor’s

1. Introduction

Shor's quantum algorithm for integer factorization has been expected to defeat RSA eventually, with only technical hurdles such as increasing the number of qubits ahead. However, we will consider aspects of the algorithm that leave open a classical explanation for physical implementations of Shor's with very small modulus. We will end by suggesting a direction to avoid the loophole in future experiments.

2. Analysis

The quantum part of Shor's Algorithm consists of a loop that terminates when a certain modular exponentiation equals one. If the value immediately output from a quantum operation does not produce this result, various related values are tried, such as multiples. Failing that, the quantum operation may also be retried. A similar pattern of a loop preventing wrong computed values from exiting together with a retry of the operation with, critically, a changed input parameter to the computation, also applies to the overall algorithm that includes the quantum part. We may summarize this uneloquently as “keep changing things until you get the right answer.” Of course, such a scheme may be implemented classically as well, leaving a loophole in experiments. In the next section, Python code implementing a version of Shor's Algorithm is given in which all the math not in loop conditions or 'if' statements is replaced by random number generation. Except for an occasional infinite loop, it produces the factors of an input integer. By the mechanism above, Shor's Algorithm implementations may resemble a classical brute force search unintentionally for very small moduli.

3. Analogous Classical Code and Output

```
---- begin code
import random
from fractions import gcd

n = 23 * 41

def modexp(x, e, n):
   y = 1
   while e > 1:
      if (e % 2 == 1):
         y = x * y
         y = y % n
      x = x * x
      x = x % n
```

```
        e = e // 2
    return (x * y) % n

a = 0
r = 0
while True:
    a = random.randint(1, n-1)
    r = random.randint(1, n-1)
    while(modexp(a, r, n) != 1):
        r = random.randint(1, n-1)
    if(r % 2 == 1):
        continue;
    if(modexp(a, r/2, n) == n - 1):
        continue;
    factor1 = gcd(modexp(a, r/2, n)+1, n)
    factor2 = n // factor1
    if (factor1 != 1 and factor2 != 1):
        break;
print("factors:")
print(factor1)
print(factor2)
----end code

---- Output:
>Shor.py
factors:
41
23
```

4. Closing the Loophole

The number of times that the loops execute should be able to reveal the difference between classical and quantum operation. Essentially, the classical case will consist of many executions and the quantum case will consist of few. Studies of future Shor’s algorithm implementations should compare the number of loop iterations between a random-number-generation-based implementation such as that presented above and the hardware Shor’s implementation, with a statistically adequate number of trials, and with various moduli if applicable. The difference between the cases will increase with increasing modulus because the quantum case is computing and the classical case is guessing.

5. Conclusion

We presented a classical mechanism whereby a small Shor's Algorithm implementation may produce correct answers without quantum operation. The loophole is in Shor's Algorithm itself and not specific implementations, but it will be increasingly easy to distinguish the classical and quantum cases as larger numbers are factored.

References

Shor’s Algorithm. (2018, April 9). In *Wikipedia, the free encyclopedia.* Retrieved May 3, 2018, from https://en.wikipedia.org/wiki/Shor%27s_algorithm

Copyrights