SHOR'S ALGORITHMS

Stephen McAdam
Department of Mathematics
University of Texas at Austin
mcadam@math.utexas.edu

INTRODUCTION: We describe Shor's algorithms for using a quantum computer to factor an odd integer $n > 0$, not a prime power, and to solve the discrete log problem (section 6). Both algorithms are exponentially faster than any known method using a classical computer.

We assume the reader is familiar with elementary mathematics. The necessary rules about quantum physics will be given. One goal is to give the uninitiated reader a chance to contemplate the wonders of quantum physics.

SECTION 1: EXPONENTIAL FACTORIZATION.

The basic method used is a well known one. Choose a random integer b with $1 < b < n$. Use the Euclidean algorithm to find GCD(b , n). If it is greater than 1, then you have found a proper factor of n. If GCD(b, n) = 1, then find the order of b mod n, i.e., the smallest positive integer r with $b^r \equiv 1 \bmod n$. If r is odd, start again with a new b. If r is even, calculate $\text{GCD}(b^{r/2} - 1, n)$. As n does not divide $b^{r/2} - 1$, that GCD is either 1 or a proper factor of n. If it is 1, start again with a new b.

Notation: In what follows, we always assume $1 < b < n$, and GCD(b, n) = 1. Also, r will be the order of b mod n.

Definition: Let us call b useful if when we apply the above process to b, it produces a proper factor of n.

We will show that if you choose a random b with $1 < b < n$ and GCD(b, n) = 1, then the probability is at least ½ that b will be useful. Therefore, the hard part of the above process is not choosing a useful b, but is finding r, the order of b mod n. Shor's algorithm gives a way of fairly quickly producing a number r with a pretty good chance that r is the order of b mod n.

Notation: Let the prime decomposition of n be $\prod p_i^{e_i}$ $(1 \le i \le t, t \ge 2$, all primes odd). Thinking of b as a member of the group of units modulo $p_i^{e_i}$, let its order be $2^{w_i} g_i$ with g_i odd.

(1.1) Lemma: With notation as just given, b is useful if and only if $w_1, w_2, \ldots, w_t$ are not all equal.

Proof: It is easily seen that the order of b in the group of units modulo n is $r = 2^w g$ with w the maximum of the various w_i, and with g the least common multiple of the various g_i.

Suppose $w_1, w_2, \ldots, w_t$ are not all equal. Then for some i, $w > w_i \ge 0$, so r is even, as required for usefulness. Since $w_i \le w - 1$, $2^{w_i} g_i$ divides $2^{w-1} g = r/2$, and so $p_i^{e_i}$ divides $b^{r/2} - 1$. Thus, $\text{GCD}(b^{r/2} - 1, n) \ne 1$, and so that GCD is a proper factor of n.

Conversely, suppose b is useful. Then r is even and GCD($b^{r/2} - 1$, n) is a proper factor of n. Therefore, for some i, p_i divides that GCD. Since p_i is odd, it cannot also divide $b^{r/2} + 1$. Since $p_i^{w_i}$ divides $b^r - 1 = (b^{r/2} - 1)(b^{r/2} + 1)$, we see $p_i^{w_i}$ divides $b^{r/2} - 1$. Therefore, $2^{w_i} g_i$ (the order of b mod $p_i^{w_i}$) divides $r/2 = 2^{w-1}g$, showing that $w_i \leq w - 1$. As w is the maximum of all the w_j, we see they are not all equal.

Remark: The above argument shows that p_i divides GCD($b^{r/2} - 1$, n) if and only if $p_i^{w_i}$ divides that GCD, and so that GCD equals $\prod p_k^{e_k}$ over k in some subset of {1, 2, .. t}. In particular, we see why the method we are using will not factor a power of a prime.

(1.2) Corollary: A randomly chosen b with $1 < b < n$ and GCD(b, n) = 1 has probability at least $1 - (1/2)^{t-1}$ of being useful. (Since $t \geq 2$, that probability is at least ½.)

Proof: Those b can be thought of as comprising the group of units modulo n, and that group is well known to be isomorphic to the direct product of the groups of units modulo $p_1^{e_1}$ for $1 \leq i \leq t$. Furthermore, it is well known that the group of units modulo $p_1^{e_1}$ is cyclic of size $p_i^{e_i - 1}(p_i - 1)$, which we write as $2^{u_i} k_i$ (k_i odd, and $u_i \geq 1$). Suppose α_i is a generator for that group. Then that group consists of the elements $\alpha_i^{m_i}$ for $1 \leq m_i \leq 2^{u_i} k_i$. We see there is a one-to-one correspondence between the set of b we are considering and the set of all t-tuples, $(m_1, m_2, \ldots, m_t)$, as each m_i varies from 1 to $2^{u_i} k_i$. The correspondence is such that the corresponding b satisfies b mod $p_1^{e_1} = \alpha_i^{m_i}$ for $1 \leq i \leq t$. The $2^{w_i} g_i$ mentioned in the notation preceding (1.1) is therefore the order of $\alpha_i^{m_i}$, which is well known to be $2^{u_i} k_i / \text{GCD}(m_i, 2^{u_i} k_i)$. If we write $m_i = 2^{v_i} f_i$ (f_i odd), then $w_i = u_i - \min\{v_i, u_i\}$.

We summarize. Fixing some i ($1 \leq i \leq t$), we see that a random choice of b ($1 < b < n$, GCD(b, n) = 1), leads to some m_i ($1 \leq m_i \leq 2^{u_i} k_i$) which in turn gives rise to a w_i ($0 \leq w_i \leq u_i$).

We now find the probability that the randomly chosen b will lead to w_i equaling u_i. To get $w_i = u_i$, we need $v_i = 0$, which is to say we need m_i to be odd. That is true of exactly half the m_i between 1 and $2^{u_i} k_i$, and so of half the possible b (via the above one-to-one correspondence). Thus, there is probability ½ that a random choice of b produces $w_i = u_i$. It trivially follows that for any fixed integer d, there is probability at most ½ that the random b leads to $w_i = d$.

Choose a random b from our set, and consider the corresponding t-tuple $(w_1, w_2, \ldots, w_t)$. We claim that the probability that all its entries are equal is at most $(½)^{t-1}$. Suppose $w_1 = d$. We have just seen that for $2 \leq i \leq t$, the probability that w_i also equals d is at most ½. Therefore, the probability that $(w_1, w_2, \ldots, w_t) = (d, d, \ldots, d)$ is at most $(½)^{t-1}$, proving the claim. The corollary now follows from (1.1).

(1.3) Lemma: $r \leq \phi(n)/2 < n/2$, (with ϕ the Euler phi function). (Since we do not know the factors of n, we do not know $\phi(n)/2$.)

Proof: It is well known that the group of units mod n is not cyclic. Thus no element has order $\phi(n)$. Since r divides $\phi(n)$, the lemma follows.

SECTION 2: QUANTUM COMPUTERS.

PREVIEW: Let us think of a computer as a black box having an input slot, and an answer slot. We put the data, and the instructions concerning what we want the machine to do with that data, into the input slot. We read the answer from the answer slot.

The input slots of classical and quantum computers behave (for our purposes) essentially the same way. However, there are profound differences in the behavior of their answer slots. The answer slot of a classical computer will contain one definite answer, A, (predetermined by what went into the input slot). When we read that answer, we do not change it; the content of the answer slot is not affected by our reading it. Also, if we want, we can arrange for the machine to make many copies of A, putting each in its own answer slot. We can do that before reading the answer, so that copies A are abundant (even if we do not yet know what A is).

None of that is true for the answer slot of a quantum computer. Instead of containing one predetermined answer, it will (usually) contain many potential answers, A_k (k in some index set). What answer we actually do read will be determined in part by the physical act of the reading. Each potential answer A_k is tagged with a probability $P(A_k)$, which is the probability that the act of reading will produce the answer A_k. When we do the reading, and get one of the potential answers A_k, the content of the answer box instantly changes; all the other potential answers are banished, and A_k alone remains. $P(A_k)$ instantly becomes 1, so that if read again, the answer slot will again give A_k. What that means is that the other potential answers that were not read on our first try, are lost to us. Any useful information they might have contained has vanished. Although before the first reading, the answer slot contained a choice of potential answers, we only get to read one of them (and chance determines which one, so that we do not get a choice). We could try to trick nature, and try to make copies of the answer slot before reading it. If we could do that, we would have many answer slots. Reading them one by one, we would get answers A_k, A_j, A_h, etc, and hopefully, some of them would be useful. Alas, that is not possible. The laws of quantum physics show that we cannot make any copies of the answer slot (until after we have read it, when copying is too late to be useful). That is called the 'no cloning theorem'. (See [G] or [M1].)

In short, while a quantum computer will tantalizingly have many potential answers hidden inside it, it will only tell us one of them, and which one it lets us read is determined by chance. Therefore, the trick, which Peter Shor successfully pulled off, is to arrange matters so that $P(A_k)$ is big when A_k is useful, making it more likely that the answer we read will be a useful one.

We will work with the polarization states of photons, although an alternative is the spin states of electrons. The reason that quantum computers do not yet exist, except as prototypes, is that photons and electrons are hard to manipulate. However, progress is slowly being made. Some people are trying to develop quantum computers based on more manageable technology (larger objects). However, the mathematics is pretty much all the same, and so photons will suffice for our purposes.

The magnetic field of a typical photon oscillates in a plane, while its electric field oscillates in a perpendicular plane. (The line of intersection of the two planes is the line of the photon's travel.) We will discuss the magnetic field. When we say a photon is polarized horizontally (respectively, vertically), we will mean its magnetic field lies in

the horizontal (respectively, vertical) plane. We will use the symbol $|0\rangle$ (respectively $|1\rangle$) to denote a horizontally (respectively, vertically) polarized photon.

Suppose the magnetic field of a photon lies in a plane at angle θ to the horizontal. We will use the symbol $\cos\theta|0\rangle + \sin\theta|1\rangle$ to denote such a polarization state. In general, all polarization states can be denoted by $a|0\rangle + b|1\rangle$, with $|a|^2 + |b|^2 = 1$ (and conversely, all such expressions correspond to states of actual photons).

In the previous parargraph, a and b are complex numbers (and $|x|^2 = x^*x$, with x^* the complex conjugate of x). The need for complex numbers is because there exist photons whose magnetic fields are not confined to a single plane. For instance, a circularly polarized photon has a magnetic field that constantly changes angle, going round and round a circle. Such a photon is described by the state $(1/\sqrt{2})|0\rangle \pm (i/\sqrt{2})|1\rangle$ (the ± because there are two ways to go around a circle).

What we have described above are the unit vectors in 1-qubit space, by which we mean the 2-dimensional complex inner product space having the symbols $|0\rangle$ and $|1\rangle$ as an orthonormal basis. The inner product of $a|0\rangle + b|1\rangle$ and $c|0\rangle + d|1\rangle$ (in that order), is $a^*c + b^*d$, so that $a|0\rangle + b|1\rangle$ is a unit vector exactly when $|a|^2 + |b|^2 = 1$.

We will have a device by which we can 'read' a photon. When the device is applied to the photon, the device registers either a 0 or a 1 (the reading). Concerning the nature of the device, we will only say it is not unlike the lens in polariod sunglasses. Here is a rule (law) of quantum physics (verified by experiment).

PROBABILITY RULE: When a photon in state $a|0\rangle + b|1\rangle$ is read, the probability of getting 0 is $|a|^2$ and the probability of getting 1 is $|b|^2$. (Recall, $|a|^2 + |b|^2 = 1$.)

Recall, in our preview of quantum computers, we said that the answer slot will contain a collection of answers, and probability will determine which answer the machine gives us. That stems from the above rule. Suppose the computer is small, having only one photon in its answer slot. Suppose that photon is in state $a|0\rangle + b|1\rangle$, with $a \neq 0 \neq b$. Then the answer slot can be thought of as containing both the answers 0 and 1 in what is called a super-position of states. When we read the answer, we might get either 0 or 1, with probabilities $|a|^2$ and $|b|^2$ respectively.

The probabilities mentioned above appear to arise from pure chance. The outcome of flipping a coin seems to depend on chance, but only because the relevant variables (size and weight of the coin, position of thumb, force of flip, wind currents, etc) are hard to determine with enough accuracy to predict the outcome. However, when reading a photon, the best current understanding of physics strongly indicates that there are no variables involved at all. Chance, and only chance determines the outcome. Most physicists now accept that, although historically, many had trouble accepting it. (Einstein never did accept it.)

We also said that reading the answer slot of a quantum computer changes the contents of the answer slot. That follows from the next rule.

CONVERSION RULE: If a photon in state $a|0\rangle + b|0\rangle$ is read and gives 0, then the photon converts to state $|0\rangle$. If the reading is 1, the photon converts to state $|1\rangle$.

Suppose the photon in the answer slot is initially in state $a|0> + b|1>$. Suppose upon being read, it gives 0. Then the photon converts to state $|0> = 1|0> + 0|1>$. If we read it again, the probability rule shows the probability of reading 0 is 1, a dead certainty. Thus, if we get a 0 on our first reading, all later readings also give 0. The potential for getting 1 (which existed before the first reading) has vanished. A similar statement holds if the initial reading is 1.

In [S], Shor says "… computer scientists have tended to forget that computation is dependent on the laws of physics." When dealing with a quantum computer, it is harder to forget. Going one step further, all information must be stored in a physical medium, be it engraved tablets or brain neurons. 'Pure' thought does not exist independent of the physical world. Interestingly, there are some physicists who seem to treat the physical world as a collection of information.

We need to discuss the behavior of the polarization state of a collection of many photons, and we need convenient notation.

NOTATION (part I): We will introduce useful notation via example. Suppose we have two photons, the first in state $|1>$ and the second in state $|0>$. It is convenient to say that together, the two photons are in state $|10>$. The symbols $|00>$, $|01>$, and $|11>$ have similar interpretations. When dealing with two photons, we will be working in 2-qubit space, the 4-dimensional complex inner product space having the symbols $|00>$, $|01>$, $|10>$, $|11>$ as an orthonormal basis. When the integers 0 through 3 are expressed in base 2, we have 00, 01, 10, and 11. Therefore, we will shorten $|00>$, $|01>$, $|10>$, $|11>$ by instead writing $|0>$, $|1>$, $|2>$, $|3>$. (The symbol $|0>$ by itself is ambiguous, but knowing we are in 2-qubit space, we know it means $|00>$.) Similarly, three photons will use 3-qubit space (2^3-dimensional) having orthonormal basis $|0>$, $|1>$, $|2>$, $|3>$, $|4>$, $|5>$, $|6>$, $|7>$, where, for instance, $|5>$ is shorthand for $|101>$. For m photons, we are in m-qubit space, having orthonormal basis $\{ |j> \mid 0 \le j \le 2^m - 1\}$. Of course, a unit vector in m-qubit space has the form $\sum c_j|j>$ where $\sum|c_j|^2 = 1$ for $0 \le j \le 2^m - 1$. This base 2 shorthand will be used when it is convenient. (It is the method used to encode integers for use by the computer. If you want to feed the number 2 into the input slot, you stick in two photons, the first polarized vertically--$|1>$-- and the second polarized horizontally--$|0>$-- together giving $|10>$, i.e. $|2>$.)

Suppose we have an ordered collection of photons, and we read each of them. Each photon will either give a reading of 0 (in which case it will convert to state $|0>$) or it will give a reading of 1 (in which case it will convert to state $|1>$). Specifically, suppose we have three photons, and when they are read, the first reads as 1, the second reads as 0, and the third reads as 1. Together, they give the 3-tuple (1, 0 ,1). Again thinking base 2, we will say the 3 photons give a reading of 5. We also know that the first photon converted to state $|1>$, the second converted to state $|0>$, and the third converted to state $|1>$. Therefore, we denote the new state of the three photons as $|101>$, or for short, $|5>$. In general, if a collection of m photons, is read as j, then upon being read, they convert to state $|j>$. Here, the allowable j are $0 \le j \le 2^m - 1$.

FUNDAMENTAL RULE: If a collection of photons in state $\sum c_j|j>$ is measured, the probability that it will read as j is $|c_j|^2$, and if that is the case, the collection converts to state $|j>$.

NOTATION (part II): In our application, we will have a collection of m + k photons, thought as a first collection of size m and a second collection of size k. Let us modify the above notation appropriately. Again, we work via example, considering four photons thought of as a first pair of photons and a second pair of photons. Suppose the four photons together are in state |1101>. Thus, the first pair is in state |11> (shorthanded as |3>) while the second pair is in state |01> (shorthanded as |1>). We will write the state of the two pairs of photons as |3, 1>. Similarly, |2, 2> is our new shorthand for |1010>.

Consider the set B = { |j, h> | $0 \le j \le 2^m - 1$, $0 \le h \le 2^k - 1$}. That set is the base 2 shorthand form of the canonical orthonormal basis of (m + k)-qubit space. (Example: For m = 3, k = 2, we have that |6, 3> is shorthand for the first 3 photons being in state |110> and the last two photons being in state |11>, so together, the five photons are in state |11011>, which is one of the 32 vectors in the canonical basis of 5-qubit space.)

PRODUCT STATES: The notation (part II) discusses a collection of m + k photons. Suppose the first m photons are in the m-qubit state $\sum_{j=0}^{2^m-1} c_j \mid j >$ and the last k photons are in the k-qubit state $\sum_{h=0}^{2^k-1} d_h \mid h >$. What (m + k)-qubit state describes the two collections thought of as a single system? We claim the answer is $\sum c_j d_h \mid j,h >$ ($0 \le j \le 2^m - 1$, $0 \le h \le 2^k - 1$), which is called the tensor product of the two initial states. The entire proof is slightly more elaborate than we wish to do here. (See [M1, section 2].) We will give a partial proof, showing that the product state correctly describes the result of taking a reading of our m + k photons.

If we take a reading, the fundamental rule shows the probability the first m photons will read as j is $|c_j|^2$, and if so, they go into state |j>. Similarly, the probability is $|d_h|^2$ that the last k photons read as h, and if so, they go into state |h>. Thus, there is probability $|c_j|^2|d_h|^2$ that the reading will be (j, h), and that the final state will be |j, h>. Now, if we have m + k photons in state $\sum c_j d_h \mid j,h >$, the fundamental rule says that upon being read, there is probability $|c_j d_h|^2$ that the reading will be (j, h), and if so, the photons will go into state |j, h>. That agrees with the preceding, and completes our partial proof. (A complete proof would have to take quantum gates into account. See remark 2.1 below.)

We will only need a special case of product states, which we now examine. Suppose we have two photons, both in the 1-qubit state $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$. Taken together, they are in the product state (1/2)|0, 0> + (1/2)|0, 1> + (1/2)|1, 0> + (1/2)|1, 1>. Now |1, 1> means |11>, and in shorthand, that is |3>. Thus, the two photons are in the product state $(1/\sqrt{2^2}) \sum_{j=0}^{3} \mid j >$. Now suppose we have two more photons, both in 1-qubit state |0>, so together in 2-qubit state |00> = |0>. Our four photons taken all together will be in the tensor product of $(1/\sqrt{2^2})$ $\sum_{j=0}^{3} \mid j >$ and |0>. It is easily seen that product is $(1/\sqrt{2^2}) \sum_{j=0}^{3} \mid j,0 >$.

At one step of our process, we will have m + k photons, each of the first m being in

1-qubit state $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$, and each of the last k being in 1-qubit state |0>. Taken together, the m + k photons will be in the (m + k)-qubit state $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} |j,0>$.

PARTIAL READINGS & ENTANGLED STATES: In our application, we will have m + k photons. At one step, we will do a partial reading, by only reading the last k photons. We need to understand what transpires. Again, we illustrate with two examples, the first rather boring, but the second quite interesting. (We mention that the fundamental rule does not require total readings to read all the photons simultaneously. You are allowed to read some now, and the rest later. The fundamental rule still applies.)

Example A: Suppose m = 2 and k = 1, and suppose the three photons are in the 3-qubit state $(1/\sqrt{2^2}) \sum_{j=0}^{3} |j,0> = (1/\sqrt{2^2})[|0, 0> + |1, 0> + |2, 0> + |3, 0>]$. We recognize this as the product of the 2-qubit state $(1/\sqrt{2^2}) \sum_{j=0}^{3} |j>$ and the 1-qubit state |0>.

If we take a total reading, the four possible outcomes are (0, 0), (1, 0), (2, 0), and (3, 0), all equally likely. Therefore, if we instead had just read the first pair of photons, we could get any of 0, 1, 2, or 3, all with equal probability.

However, now let us suppose that we first read the third photon, and later read the first pair of photons. Does that change the possibilities of the outcomes from the first pair? In this example, the answer is no. Reading the third photon must produce 0, and that does not eliminate any of the four possible total readings. Thus, if we later read the first pair of photons, we again see the possible outcomes 0, 1, 2, or 3 can occur with equal probability.

In this example, reading the third photon has no influence on the result of reading the first pair of photons. (That is the nature of product states.) That is not the case in the next example.

Example B: We again take m = 2, k = 1, but suppose the state of the 3 photons is $(1/\sqrt{2^2})[|0, 0> + |1, 0> + |2, 0> + |3, 1>]$.

A total reading has each of the possible outcomes (0, 0), (1, 0), (2, 0), or (3, 1) all being equally likely. Thus, if we first read the first pair of photons, the possibilities 0, 1, 2 or 3 must all be equally likely.

However, suppose we instead first read the last photon, and get 1. That means when we later complete the reading, by reading the first pair of photons, that pair must give us 3. What that means is that our act of reading the last photon (and getting 1), causes the first pair of photons to go into state |3> (since when read, they *must* give 3).

Similarly, if we read the last photon and get 0, we have eliminated the possibility of getting (3, 1) from the total reading. Therefore, when we later read the first pair of photons, they can only give us 0, 1, or 2. A simple 'conditional probability' argument shows each of those outcomes is equally likely, having probability 1/3. Therefore, our act of reading the last photon (and getting 0), causes the first pair of photons to go into the state $(1/\sqrt{3})[|0> + |1> + |2>]$.

In the previous example (a product state), reading the last photon did not influence a later reading of the first pair of photons. In this example, reading the last photon does influence a later reading of the first pair of photons, restricting what the outcomes can be. In this example, the first pair of photons and the last photon are in what is called an entangled state.

Entangled states are arguably the weirdest of all the weird things in quantum physics. Let us briefly explain why. The rules do not require that the last photon be near the first pair of photons when the readings are done. Theoretically, they can be separated by light years. And yet, reading the last photon has an essentially instantaneous effect on the far distant pair of first photons. The effect must be accomplished without the communication of any information, since otherwise, we would have faster than light communication. Einstein referred to this phenomenon as 'spooky action at a distance', and never accepted its existence. Yet modern physicists (with a few exceptions) routinely accept it. Experiments over tens of kilometers, using very sophisticated equipment, confirm the existence of such spooky action. For more, see [M1] and [M2].

QUANTUM GATES: Data in a classical computer is manipulated via logic gates. In a quantum computer, data (encoded in polarization states) is manipulated via quantum gates. A quantum gate changes the state of a collection of photons (without reading the collection). If one reads a collection of photons, the reading changes their state to a known state (known via the readout). However, if one applies a quantum gate to a collection of photons in an unknown state, that state will change to some other unknown state.

Quantum gates act linearly! If a collection of photons in state $\sum c_j|j>$ is passed through a quantum gate represented by G, the resulting state $G(\sum c_j|j>)$ will equal $\sum c_j G(|j>)$. Therefore, to understand the workings of a quantum gate in m-qubit space, we need only understand how that gate affects the orthonormal basis $\{ |j> \mid 0 \le j \le 2^m - 1\}$. That is easily described, since quantum gates correspond to unitary operators.

Definition: A unitary operator on m-qubit space is a linear transformation from m-qubit space to itself that carries the orthonormal basis $\{ |j> \mid 0 \le j \le 2^m - 1\}$ to some orthonormal basis. (It is easily seen that a unitary operator carries a unit vector to a unit vector.)

QUANTUM GATE RULE: There is a correspondence between quantum gates acting on m photons, and unitary operators acting on m-qubit space. If a quantum gate corresponds to the unitary operator G, then $\{G(|j>) \mid 0 \le j \le 2^m - 1\}$ is an orthonormal basis of m-qubit space. Also, if a set of m photons in state $\sum c_j|j>$ is passed through that gate, then those photons will convert to state $G(\sum c_j|j>) = \sum c_j|G(|j>)$. (The question of whether every unitary operator represents an actual quantum gate appears to be a bit subtle. It seems that with some physical effort put into construction, it is approximately true. The gates we mention in this work all exist.)

Remarks: a) When dealing with polarization states of photons, there are only two things you can do. You can apply quantum gates, and you can take readings. Thus, a quantum computer must make do with those two tools.

b) Some quantum gates require more individual steps (subgates) to implement then others do. I have read that in general, if you have N photons, any gate can be implemented using at most $O(2^{2N})$ steps. That is not good enough for us. The gates we need can be implemented with far fewer steps.

c) Given states α and β, there are many unitary operators carrying α to β, and so many quantum gates which convert the state α into the state β. However, since we want to be able to know that the gates we use are all fast, an existence statement is not enough. We must specify the gates we are using.

Here is our immediate (but not final) goal. We wish to start with m + k photons, all in the 1-qubit state |0>, and so together in the (m + k)-qubit state |0, 0> (using shorthand). We also have a function g from $\{j \mid 0 \le j \le 2^m - 1\}$ to $\{h \mid 0 \le h \le 2^k - 1\}$. We wish to show that quantum gates can be used to convert |0, 0> to the (m + k)-qubit state $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} |j, g(j)>$. We will do that in two steps.

HADAMARD GATE: In 1-qubit space, the unit vectors $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$ and $(1/\sqrt{2})|0> - (1/\sqrt{2})|1>$ are an orthonormal basis, (orthogonal since $(1/\sqrt{2})*(1/\sqrt{2}) + (1/\sqrt{2})*(-1/\sqrt{2}) = 0$). Therefore, the linear transformation sending |0> to $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$ and |1> to $(1/\sqrt{2})|0> - (1/\sqrt{2})|1>$ is a unitary operator, which corresponds to a quantum gate called the Hadamard gate, denoted H.

The exercises we give are not required, but are recommended.

Exercises: a) Show H is its own inverse.

b) Show $H(\cos\theta|0> + \sin\theta|1>) = \sin(\theta + \pi/4)|0> + \cos(\theta + \pi/4)|1>$.

c) Suppose there are two large collections of photons, the first all in state $(1/\sqrt{2})|0> + (1/\sqrt{2})|1> = \cos(\pi/4)|0> + \sin(\pi/4)|1>$ (polarized at angle $\pi/4$), and the second all in state $(1/\sqrt{2})|0> + (i/\sqrt{2})|1>$ (circularly polarized). Suppose you read all the photons in the first collection. The fundamental rules shows that half the readings will be 0 and half will be 1. The same is true of the second collection. Therefore, you cannot distinguish between the two collections by just reading them. Show that if instead, you first pass all the photons in both collections through the gate H, and then read the results, you can distinguish between the two collections.

d) Suppose c is a complex number with $|c|^2 = 1$ (such c being called phase factors). Suppose you have two collections of photons, the first all in some fixed state α, and the second all in state $c\alpha$. Show that there is no way to distinguish between them. (Recall, your only tools are quantum gates and readings.) For that reason, α and $c\alpha$ are considered the same state.

(2.1) Remark: Part (d) of the previous exercise shows that there is no way to distinguish between a photon in state α and a photon in state $c\alpha$, when c is a phase factor. Similarly, it can be shown that if you have m photons in state $\sum_{j=0}^{2^m-1} c_j \mid j >$, and k photons in state $\sum_{h=0}^{2^k-1} c_h \mid h >$, taken together, they cannot be distinguished from m + k photons in the tensor product state $\sum c_j d_h \mid j,h >$. (See [M1, section 2].)

Let H be the Hadamard gate. Suppose we have a pair of photons both in the 1-qubit state |0>, (and so as a pair, in the 2-qubit state |00>). Suppose we apply H to both of the photons (that operation constituting a quantum gate we denote $H^{\otimes 2}$). The two photons then both go into the 1-qubit state $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$. Together, they are in the 2-qubit (product) state $(1/2)|00> + (1/2)|01> + (1/2)|10> + (1/2)|11> = (1/2)|0> + (1/2)|1> + (1/2)|2> + (1/2)|3> = (1/\sqrt{2^2}) \sum_{j=0}^{3} \mid j >$

In general, beginning with m photons in the m-qubit state |000….00>, we can similarly apply H to each of them, and convert them to state $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} \mid j >$. Using |0> as shorthand for |000…00>, and using $H^{\otimes m}$ to denote passing each of the m photons through the Hadamard gate, we get $H^{\otimes m}(|0>) = (1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} \mid j >$.

Remark: The unit vectors $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$ and $(-1/\sqrt{2})|0> + (1/\sqrt{2})|1>$ are an orthonormal basis of 1-qubit space, and so if H′ send |0> to $(1/\sqrt{2})|0> + (1/\sqrt{2})|1>$ and |1> to $(-1/\sqrt{2})|0> + (1/\sqrt{2})|1>$, then H′ is a unitary operator. It is easily seen that $H'^{\otimes m}(|0>) = (1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} \mid j >$, and so we could have used H′ instead of H. However, all the literature I have seen uses H. I do not know why. Perhaps H is easier to physically implement than H′. It is easily seen that H′ rotates a linearly polarized photon $\pi/4$ radians counter-clockwise, and so is easier to envision than H.

Now suppose we have m + k photons, all initially in the 1-qubit state |0>. Together, they are initially in state |0, 0> (the product of the m-qubit state |0> and the k-qubit state |0>). Suppose we apply $H^{\otimes m}$ to the first m photons, but leave the last k photons alone. The result is the product of the m-qubit state $H^{\otimes m}(|0>) = (1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} \mid j >$, and the k-qubit state |0>.

We previously saw that product equals the (m + k)-qubit state $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} \mid j,0 >$.

We are half way to our goal, having seen how to convert |0, 0> to $(1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,0>$. We now wish to find a quantum gate that converts $(1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,0>$ to $(1/\sqrt{2^m})$ $\sum_{j=0}^{2^m-1}|j,g(j)>$. Recall $g : \{j \mid 0 \le j \le 2^m - 1\} \rightarrow \{h \mid 0 \le h \le 2^k - 1\}$.

Notation: As is standard in cryptography, we use $\oplus$ to denote bit-wise addition modulo 2. For example, $101101 \oplus 110111 = 011010$. Note that $X \oplus X$ is a string of 0s. Thus $(Y \oplus X) \oplus X = Y$.

Recall that the set $B = \{ |j, h> \mid 0 \le j \le 2^m - 1, 0 \le h \le 2^k - 1\}$ is the canonical orthonormal basis of (m + k)-qubit space. The map sending |j, h> to $|j, h \oplus g(j) >$ is easily seen to be a permutation of B. Since B is an orthonormal basis, we can use linearity to extend that map to a unitary operator on (m + k)-qubit space. That operator corresponds to a quantum gate, which we call S. We have that S converts the state |j, h> to the state $|j, h \oplus g(j) >$. In particular, $S(|j, 0>) = |j, g(j) >$. By linearity, $S((1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,0>) = (1/\sqrt{2^m})$ $\sum_{j=0}^{2^m-1} S(|j,0>) = (1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,g(j)>$.

In summary, we started with m + k photons in the state |0, 0>, first used m copies of the H gate to convert that to the state $(1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,0>$, and then used the S gate to arrive at the state $(1/\sqrt{2^m})\sum_{j=0}^{2^m-1}|j,g(j)>$.

In general, implementing S will involve many steps, and will be slow. However, in our application, g will be the function taking j to b^j mod n. (Here, we follow cryptography usage, and b^j mod n will mean the remainder upon dividing b^j by n, so that b^j mod n is an integer.) For this special g, the gate S is fast enough for our needs (but apparently is still the slowest step in the whole process).

Since we need $g(j) = b^j$ mod n to be in the set $\{h \mid 0 \le h \le 2^k - 1\}$, we take k such that $n < 2^k$. Increasing k increases the number of photons the computer must deal with, and so slows things down. Therefore, we take $n < 2^k < 2n$.

SECTION 3: QUANTUM FOURIER TRANFORM.

We are now ready to explain the most important step of what Shor did. We will have m + k photons with $n < 2^k < 2n$ and either $n^2/4 < 2^m < n^2/2$, or $n^2 < 2^m < 2n^2$. The first option assures a nonzero probability of success on any given attempt (a reasonably high probability, in fact). In the second option, the odds of success slightly more than double, but the process takes longer since more photons are used. We will consider both options. (The best compromise might well be to assume $n^2/2 < 2^m < n^2$, but that involves two cases, and is awkward to present. We will treat it in later exercises.)

The initial input to our quantum computer will be m + k photons in the state |0, 0>. The first two steps in the 'program' will instruct the computer to apply H to each of the first m photons, and then apply S to the result. At that stage in the proceedings, the m + k photons will be in the state $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} |\, j, b^j \bmod n >$.

Example: This will be a developing example. We will return to it several times, as our knowledge grows. In this example, we will use the $n^2 < 2^m$ option. We take n = 21 and b = 2. With m = 9 and k = 5, we have $n^2 < 2^m = 512 < 2n^2$ and $n < 2^k < 2n$. Thus our computer will use 9 + 5 = 14 photons. We load them into the machine in the initial state |00000000000000> = |0, 0>. After applying the two steps described above, the 14 photons are in state $(1/\sqrt{512})$ $\sum_{j=0}^{511} |\, j, 2^j \bmod 21 > = (1/\sqrt{512})\big[|0, 1> + |1, 2> + |2, 4> + |3, 8> +$
$|4, 16> + |5, 11> + |6, 1> + |7, 2> + |8, 4> + |9, 8> + |10, 16> + |11, 11> + |12, 1> + \ldots\ldots +$
$|504, 1> + |505, 2> + |506, 4> + |507, 8> + |508, 16> + |509, 11> + |510, 1> + |511, 2>\big]$.
(Observe that is an entangled state. If we read the last k photons and get, say 8, then a later reading of the first m photons must give a j such that $2^j \bmod 21 = 8$. Spooky action plays a role in what we are doing.)

The above state will be hidden from us. If we want to extract information, we must take a reading. If we take a total reading, we get one of the $(j, b^j \bmod n)$, each having probability 1/512 of appearing. For example, we might get $(9, 2^9 \bmod 21) = (9, 8)$. That is useless to us. (If we were lucky enough to get a reading such as (504, 1), it would tell us 504 is a multiple of r, the order of 2 mod 21, which might be of some use, since it is r we seek. However, for r large, the odds of being that lucky are small.)

Remark: A classical computer can certainly be programmed to find the first j > 0 with $b^j \bmod n = 1$ (so that j is the r we seek). However, for large n and r, that will take a lot of time, since each $b^j \bmod n$ requires a separate calculation. In a quantum computer, all the $b^j \bmod n$ are calculated simultaneously. Simply passing our photons through the gates H^m and S described above accomplishes that, making things much faster. That is the power of quantum computing. However, the previous example tells the bad news. The computer quickly finds all the information we could possible want, but it only lets us see one piece of information, and we cannot even pick which piece.

Instead of taking a total reading, we take a partial reading of the last k photons. (At the end of this section, we will explain that taking this partial reading is not really necessary. However, it does simplify notation, by focusing our attention on a useful pattern.)

Example: Continuing the previous example, we read the last 5 photons. Suppose we get 8. As described in the section on partial readings (example B), the photons now go into state

$(1/\sqrt{85})[|3, 8> + |9, 8> + |15, 8> + \ldots. + |501, 8> + |507, 8>]$. The quick way of finding the coefficient $1/\sqrt{85}$ is to note that we have a sum of 85 orthogonal unit vectors. The norm of that sum is therefore $\sqrt{85}$, and so we normalize that sum by the coefficient $1/\sqrt{85}$.

However, this is still hidden from us. If we simply read the first m = 9 photons, we will again get something like (9, 8), which is still useless. We need to find the length of the gaps between the numbers 3, 9, 15, …, 501, 507, since that length is r, the order of 2 mod 21.

NORMALIZATION: Suppose when we read the last k photons, instead of getting 8, we had gotten 2. Then the normalizing coefficient would have been $1/\sqrt{86}$. That is because of the final term |511, 2>, part of an incomplete cycle. It is easily seen that if $2^m = rf + s$, with $0 \le s < r$, then that normalizing coefficient is either $1/\sqrt{f}$ or $1/\sqrt{f+1}$ (depending on chance). We do not care which. Therefore, we will simply call that coefficient $1/\sqrt{C}$ (with C either f or f + 1).

We note a few facts used later. We know r < n/2, and we have $n^2/4 < 2^m$ (in both options we are considering). Thus $n^2/4 < rf + s < r(f + 1) \le rC$. Therefore, $C > n^2/4r > n^2/(4(n/2)) = n/2$. In particular, C is very big, as is f (under the realistic assumption that n is very big).

Using that $rf \le 2^m = rf + s < r(f + 1)$ and $f \le C \le f + 1$, we easily see $f/(f + 1) \le rC/2^m \le (f + 1)/f$. Knowing f is very big, we see $|rC/2^m - 1|$ is very small. Since $r \ge 1$, that shows $|C/2^m - 1/r|$ is very small.

At this point, we no longer need the last k photons. After all, they have already been read, and have gone into some definite state—in the above example, state |8>. We dismiss them, and just keep the first m photons. In the example, they are in state $(1/\sqrt{85})[|3> + |9> + |15> + \ldots. + |501> + |507>]$.

In general, if the last k photons read as h, and if $d \ge 0$ is the smallest integer such that $b^d \bmod n = h$, then at this stage in the process, we will have the m-qubit state $(1/\sqrt{C})$ $\sum_{u=0}^{C-1} |d + ur>$, with C either f or f + 1 (as above), and with r the order of b mod n.

Before explaining what is done next, let us mention something that we (regrettably) cannot do. In the preview of quantum computing, we mentioned that we only get to do one reading, and that reading destroys all other information that was not read. Suppose that were false. Suppose we could take multiple readings of the state $(1/\sqrt{85})[|3> + |9> + |15> + \ldots. + |501> + |507>]$. The first might give 9 and the second might give 135. That would tell us that 135 – 9 = 126 is a multiple of r (= 6). A third reading might give 225, telling us 225 – 135 = 90 is a multiple of r. We now see r divides GCD(90, 126) = 18. Continuing in this way, we could significantly narrow down the possibilities for r. Alas, quantum physics does not allow that. The most we could do is get one (and only one) number from the list 3, 9, 15, …., 501, 507.

EXERCISE: Explain why iterating will probably not help. That is, suppose at the first iteration we read the last k photons as 8, and then read the first m photons as 9 (from the list 3, 9, 15, … 501, 507). Now suppose we iterate, beginning from the start (with the same b), do the first two

steps of the program as explained above, and then read the last k photons as y and then the first m photons as x. Why is it likely that 9 and x together are of no help in finding r? Why is y = 8 the only case in which x and 9 together might help us find r (unless x also equals 9). Show that the odds that y = 8 is approximately (but not exactly) 1/r = 1/6. In this small example (n = 21, b = 2, r = 6) iteration would help, since 1/r is fairly large. (Of course, when r is small, a brute force search for it is feasible.) In a more realistic example, with n very large, the odds are that r will be very large.

EXERCISE: Let n = pq with p and q distinct large primes. Show that there are $\phi(p-1)\phi(q-1)$ choices of b with $1 \le b \le n$, such that b is a primitive root of both p and q, and that for any such b, r = LCM(p – 1, q – 1), which is large.

Having digressed into what does not help, we now look at what does help. We must use another quantum gate, one that implements the Fourier transform.

(3.1) Lemma: Let $\omega = e^{2\pi i/2^m}$. The set of vectors $\left\{(1/\sqrt{2^m})\sum_{t=0}^{2^m-1}\omega^{tj}\,|\,t\rangle \;|\; 0 \le j \le 2^m - 1\right\}$ is an orthonormal basis of m-qubit space.

Proof: Let j and h both be between 0 and $2^m - 1$. The inner product of $(1/\sqrt{2^m})\sum_{t=0}^{2^m-1}\omega^{tj}\,|\,t\rangle$ and $(1/\sqrt{2^m})\sum_{t=0}^{2^m-1}\omega^{th}\,|\,t\rangle$ is $(1/2^m)[(\omega^0)^*(\omega^0) + (\omega^j)^*(\omega^h) + (\omega^{2j})^*(\omega^{2h}) + \ldots + (\omega^{(2^m-1)j})^*(\omega^{(2^m-1)h})$. Using that $\omega^* = \omega^{-1}$, that sum equals $(1/2^m)\sum_{t=0}^{2^m-1}\omega^{t(h-j)}$. If j = h, that gives 1, showing the vectors in our set are unit vectors. For $j \ne h$, that sum equals $(1/2^m)\dfrac{(\omega^{h-j})^{2^m}-1}{\omega^{h-j}-1}$, (since the denominator is nonzero). Since ω is a 2^m-th root of unity, the numerator of that fraction is 0. Therefore, our vectors are orthonormal. As there are 2^m vectors in our set, they form an orthonormal basis.

We now see that the map taking the basis vector $|j\rangle$ to the unit vector $(1/\sqrt{2^m})\sum_{t=0}^{2^m-1}\omega^{tj}\,|\,t\rangle$ carries the canonical orthonormal basis of m-qubit space to an orthonormal basis of m-qubit space. By linearity, it can be extended to a unitary operator on m-qubit space. The corresponding quantum gate QF is the discrete quantum Fourier transform gate.

We apply that gate to the previous state $(1/\sqrt{C})\sum_{u=0}^{C-1}|\,d+ur\rangle$, resulting in the state

$$(1/\sqrt{C})\sum_{u=0}^{C-1}QF(|\,d+ur\rangle) = (1/\sqrt{C})\sum_{u=0}^{C-1}(1/\sqrt{2^m})\sum_{t=0}^{2^m-1}\omega^{t(d+ur)}\,|\,t\rangle =$$

$(1/\sqrt{C})\sum_{t=0}^{2^m-1} G(t)\,|t>$, where G(t) = $(1/\sqrt{2^m})\sum_{u=0}^{C-1}\omega^{t(d+ur)}$ = $(\omega^{td})(1/\sqrt{2^m})$F(t),

with F(t) = $\sum_{u=0}^{C-1}(\omega^{tr})^u$.

This is our final state, waiting to be read. Since it is still a unit vector (the gate QF did not change that fact), the probability of reading t is (by the fundamental rule), $|G(t)|^2/C = |F(t)|^2/2^mC$ (since $|\omega^{td}|^2 = 1$). That number is the probability that when we read the final state, we will get t. Note that the probability of getting t is not constant, but rather varies with the choice of t.

EXERCISE: Show that $|F(t)|^2 \le C^2$, so that the probability of reading t is $|F(t)|^2/2^mC \le C^2/2^mC = C/2^m \approx 1/r$. (We will soon see that for certain choices of t, the probability of reading t is at least .4/r, which is reasonably close to the permitted maximum.)

We wish to estimate the probability of reading t for certain special t, soon to be discussed. To do so, we extend the function F(t) (defined on integers t), to all real x. Thus F(x) = $\sum_{u=0}^{C-1}(\omega^{xr})^u$.

(3.2) Lemma: $|F(x)|^2 = \dfrac{1-\cos(rCx2\pi/2^m)}{1-\cos(rx2\pi/2^m)}$.

Proof: Since the sum which defines F(x) is over $0 \le u \le C-1$, we see F(x) = $\dfrac{\omega^{rCx}-1}{\omega^{rx}-1}$. (When x is an integral multiple of $2^m/r$, the denominator is 0, but so is the numerator. At those x, the definition easily shows F(x) = C, and so at those x, we assign the value C to this fraction, and C^2 to the fraction in the statement. If L'hospital's rule is applied--twice--to the fraction in the statement of the lemma at such x, it gives C^2 as the answer, and so we have continuity.) Now $|F(x)|^2 = (\dfrac{\omega^{rCx}-1}{\omega^{rx}-1})(\dfrac{\omega^{rCx}-1}{\omega^{rx}-1})^*$. That is easily seen to equal the fraction in the statement of the lemma, by using that the complex conjugate of ω is ω^{-1}, and that for real z, $\omega^z + \omega^{-z} = 2\cos(z2\pi/2^m)$ (i.e., twice the real part of ω^z).

Notation: In the interval $[1, 2^m - 1]$, the integral multiples of $2^m/r$ are are $k(2^m/r)$ for $k = 1, 2, \ldots r-1$. For each such k, let t_k be an integer with $|t_k - k(2^m/r)| \le ½$, and let $\delta_k = t_k - k(2^m/r)$ (so $|\delta_k| \le ½$). Also let P(k) be the probability that the computer gives us t_k, and let $P^{\#}(k)$ be the probability that it gives us any one of $t_k - 1$, t_k, or $t_k + 1$.

(3.3) Lemma: A lower bound for P(k) is .4/r. A lower bound for $P^{\#}(k)$ is .85/r. (Here, we assume n is very big.)

Proof: We saw above that the probability of getting t is $|F(t)|^2/2^mC$. Therefore, $P^{\#}(k) = (1/2^mC)[|F(t_k - 1)|^2 + |F(t_k)|^2 + |F(t_k + 1)|^2]$. We will show that .85/r is a lower bound for that.

For $t \in \{t_k - 1, t_k, t_k + 1\}$, write $t = k(2^m/r) + \delta$ (so that δ is one of $-1 + \delta_k$, δ_k, or $1 + \delta_k$). By (3.2), $|F(t)|^2 = \frac{1-\cos(rCt2\pi/2^m)}{1-\cos(rt2\pi/2^m)} = \frac{1-\cos((kC+(rC\delta/2^m))2\pi)}{1-\cos((k+(r\delta/2^m))2\pi)} = \frac{1-\cos((rC\delta/2^m)2\pi)}{1-\cos((r\delta/2^m)2\pi)} = \frac{1-\cos y2\pi}{1-\cos(y2\pi/C)}$, with $y = (rC/2^m)\delta$.

In the section on normalization, we noted that $|rC/2^m - 1|$ is very small. Since $|\delta|$ is small, $|y - \delta| = |rC/2^m - 1||\delta|$ will be very small, and we may use δ as a very good approximation to y. We now see $|F(t)|^2 \approx \frac{1-\cos\delta2\pi}{1-\cos(\delta2\pi/C)} \geq (2C^2)\left(\frac{1-\cos\delta2\pi}{(\delta2\pi)^2}\right)$ (using that $x^2/2 \geq 1 - \cos x$).

Since $P^{\#}(k) = (1/2^mC)\sum_{j=-1}^{1}|F(t_k + j)|^2$, we see that a very good approximation to a lower bound on $P^{\#}(k)$ is $(1/2^mC)(2C^2)\sum_{j=-1}^{1}\frac{1-\cos(j+\delta_k)2\pi}{((j+\delta_k)2\pi)^2}$.

We previously noted that $|C/2^m - 1/r|$ is very small, allowing us to approximate $C/2^m$ with 1/r**.** Therefore, our approximate lower bound becomes $(2/r)\sum_{j=-1}^{1}\frac{1-\cos(j+\delta_k)2\pi}{((j+\delta_k)2\pi)^2}$. Since we have $-1/2 \leq \delta_k \leq ½$, the minimum value of that (via graphing calculator) is slightly more than .854/r. Since our approximations improve as n grows, ($rC/2^m$ getting closer to 1), by continuity, we may take .85/r as a lower bound for $P^{\#}(k)$ when n is big.

Similarly, $P(k) = |F(t_k)|^2/2^mC$ has a lower bound which is approximated by $(2/r)\left(\frac{1-\cos\delta_k2\pi}{(\delta_k2\pi)^2}\right)$. For $-1/2 \leq \delta_k \leq ½$, the minimum value (which occurs at the endpoints) is $4/\pi^2r >$ .4/r.

(3.4) Exercise: Suppose $\delta_k \geq 0$. Show the probability of the computer giving us one of $t_k - 1$ or t_k exceeds .8/r. Suppose $\delta_k \leq 0$. Show the probability of the computer giving us one of t_k or $t_k + 1$ exceeds .8/r.

Example: We continue with our developing example. We have C = 85 and r = 6 (but we pretend we do not know those, for in reality, we would not).

The relevant integral multiples of $2^m/r = 512/6 = 85.333\ldots$ are 85.333…, 170.666…, 256, 341.333…, and 426.666…. The corresponding t_k ($1 \leq k \leq 5$) are 85, 171, 256, 341, and 427.

Consider k = 5. By (3.3), P(5) > .4/r = .4/6 = .06666…. In fact, $P(5) = (1/2^mC)|F(t_5)|^2 = (1/43520)|F(427)|^2 =$ (by (3.2)) $(1/43520)\frac{1-\cos((6)(85)(427)2\pi/512)}{1-\cos((6)(427)2\pi/512)} \approx .113$. Similarly, (3.3)

tells us $P^{\#}(5) > .85/6 = .141666\ldots$. In fact, $P^{\#}(5) = (1/43520)[|F(426)|^2 + |F(427)|^2 + |F(428)|^2] \approx .14979$.

Remark: It is illuminating to use a graphing calculator to graph $\frac{1-\cos(rCx2\pi/2^m)}{1-\cos(rx2\pi/2^m)}$ for the case r = 6, C = 85, $2^m = 512$, and $0 \le x \le 512$, and examine it near the various t_k. (It is pretty.)

Recall that the relevant k are k = 1, 2, …, r – 1. Therefore, (3.3) shows the probability that the computer gives us $t = t_k$ for some k is at worst .4(r – 1)/r, which is decently large. The probability it gives us one of $t_k - 1$, t_k, or $t_k + 1$, for some k, is at worst .85(r – 1)/r, which is quite large. The Fourier transform has successfully biased the computer's outcome so that there is a good chance we will get a t close to some $k(2^m/r)$. Why is that useful? Heuristically, it is because if $t \approx k(2^m/r)$, then $t/2^m \approx k/r$, and so r is the small (< n/2) denominator of a fraction close to $t/2^m$.

In our example, since $t_5 = 427$, (3.3) tells us the probability of our final reading being one of 426, 427, or 428 is at least .85/r = .85/6 > .14. Now each of 426/512, 427/512, and 428/512 is very close to 5/6. In fact, no fraction a/b with $1 \le b \le 21/2$ is closer to them than 5/6, leading us to guess that r might well be 6. We can easily verify that 6 is the order of b = 2 mod n = 21. Using the exponential factorization technique from section 1, we find that $GCD(2^{6/2} - 1, 21) = 7$, and so we have factored 21.

In the next section, we see exactly how to search for a fraction k/r very close to $t/2^m$, and learn that we also want GCD(k, r) = 1.

AN UNNECCESSARY STEP: We now indulge in an exercise in pedantry that can be skipped by the reader only interested in learning Shor's factoring algorithm. However, the introduction said that one goal of these notes was to help the uninitiated have a better understanding of quantum physics. The following discussion contributes to that goal.

Recall that in our example, after the first two steps of the algorithm, we read the last k photons, then applied the Fourier transform to the first m photons, and then took our final reading of the first m photons, giving us some t (hopefully near one of the t_k).

We will now explain that we did not really need to take that partial reading, but instead could have just applied the Fourier transform to the first m photons, and then take our final reading of the first m photons. We claim that the probability of getting a given t is the same, whether or not we do the partial reading. We will give two proofs of that fact. Here is the first.

Suppose we do the first two steps of the algorithm, but before doing the partial reading of the last k photons, we first move them very far away from the first m photons. (Remark: experiments show the influence between entangled photons is essentially instantaneous, regardless of how far apart the photons are.) Suppose we then read the last k photons, and then quickly apply QF to the distant first m photons, and quickly read those m photons. Special relativity tells us there will be observers for whom the reading of the first m photons will happen before the reading of the last k photons. To those folks, we will have found our t (and with a little luck, will have factored n) before the last k photons were read. So why bother reading them at all!

We now give a second, more mathematical argument, reaching the same conclusion. In general, after the partial reading of the final k photons, we saw that the if we read the first m photons, the probability of getting t is $|F(t)|^2/2^mC$, with $F(t) = \sum_{u=0}^{C-1}(\omega^{tr})^u$.

Now in our example, the reader can easily see that C = 85 in those cases where the last 5 photons are read as one of 4, 8, 16, or 11, while C = 86 in those cases where the last 5 photons are read as 1 or 2.

In our example, the probability of the last 5 photons being read as 8 is 85/512. Furthermore, if that is the case, then the *conditional* probability of reading t from the first 9 photons is $|F(t)|^2/2^mC = |\sum_{u=0}^{84}(\omega^{tr})^u|^2/(512)(85)$. The product of that probability and conditional probability is just $|\sum_{u=0}^{84}(\omega^{tr})^u|^2/(512)^2$.

On the other hand, the probability of the last 5 photons being read as 2 is 86/512. Furthermore, if that is the case, then the conditional probability of reading t from the first 9 photons is $|F(t)|^2/2^mC = |\sum_{u=0}^{85}(\omega^{tr})^u|^2/(512)(86)$. The product of that probability and conditional probability is just $|\sum_{u=0}^{85}(\omega^{tr})^u|^2/(512)^2$.

Considering all such possibilities for the partial reading of the last 5 photons, we see that the actual probability of getting t as our final reading (using the algorithm with the partial reading step) is

$$\frac{1}{(512)^2}(2|\sum_{u=0}^{85}(\omega^{tr})^u|^2 + 4|\sum_{u=0}^{84}(\omega^{tr})^u|^2).$$

Remark: It is interesting to note that if we sum the previous expression over all t with $0 \le t \le 511$, we must get 1.

Next, we will examine what happens if we leave out the partial reading of the last k photons, and show that the probability of getting t as our final reading is exactly the same.

At the end of the second step, (before taking the partial reading), the computer was in the state

$(1/\sqrt{512})[|0, 1> + |1, 2> + |2, 4> + |3, 8> +$
$|4, 16> + |5, 11> + |6, 1> + |7, 2> + |8, 4> + |9, 8> + |10, 16> + |11, 11> + |12, 1> + \ldots\ldots +$
$|504, 1> + |505, 2> + |506, 4> + |507, 8> + |508, 16> + |509, 11> + |510, 1> + |511, 2>]$.

We can rewrite that as

$$(*)\quad (1/\sqrt{512})[(\sum_{u=0}^{85}|0+6u>)\otimes|1> + (\sum_{u=0}^{85}|1+6u>)\otimes|2> +$$

$$(\sum_{u=0}^{84}|2+6u>)\otimes|4> + (\sum_{u=0}^{84}|3+6u>)\otimes|8> +$$

$(\sum_{u=0}^{84} |4+6u>)\otimes|16> + (\sum_{u=0}^{84} |5+6u>)\otimes|11>\big]$.

We previously saw that the quantum Fourier transform QF converts

$(1/\sqrt{C})\sum_{u=0}^{C-1}|d+ur>$ to $(1/\sqrt{C})\sum_{t=0}^{2^m-1} G(t)|t>$, or dropping the $1/\sqrt{C}$,

it converts $\sum_{u=0}^{C-1}|d+ur>$ to $\sum_{t=0}^{2^m-1} G(t)|t>$,

where G(t) = $(\omega^{td})(1/\sqrt{2^m})$F(t), with F(t) = $\sum_{u=0}^{C-1}(\omega^{tr})^u$.

As we need to keep track of d, we rename G(t) as G(d, t) and F(t) as F(d, t).

We recall that C is 86 when 2^d mod n is 1 or 2 (i.e., when d is 0 or 1), and C is 85 when 2^d mod n is 4, 8, 16, or 11 (i.e., when d is 2, 3, 4, or 5). Thus, for example, F(1, t) = $\sum_{u=0}^{85}(\omega^{tr})^u$, while f(3, t) = $\sum_{u=0}^{84}(\omega^{tr})^u$.

We now see that QF converts (*) into

(**) $(1/\sqrt{512})\big[(\sum_{t=0}^{2^m-1} G(0,t)|t>)\otimes|1> + (\sum_{t=0}^{2^m-1} G(1,t)|t>)\otimes|2> +$

$(\sum_{t=0}^{2^m-1} G(2,t)|t>)\otimes|4> + (\sum_{t=0}^{2^m-1} G(3,t)|t>)\otimes|8> +$

$(\sum_{t=0}^{2^m-1} G(4,t)|t>)\otimes|16> + (\sum_{t=0}^{2^m-1} G(5,t)|t>)\otimes|11>\big] =$

$(1/\sqrt{512})\sum_{d=0}^{5}\ \sum_{t=0}^{511}$ G(d, t)|t>$\otimes|2^d$ mod 21>.

If the first 9 photons are read as t, it means we tapped into a term of the form |t>$\otimes|2^d$ mod 21>, and the probability that we did tap into such a term is

$(\frac{1}{512})(|G(0,t)|^2 + |G(1,t)|^2 + |G(2,t)|^2 + |G(3,t)|^2 + |G(4,t)|^2 + |G(5,t)|^2)$.

Now using that G(d,t) = $(\omega^{td})(1/\sqrt{512})$F(d,t), and knowing how F(d, t) varies with d,

that simplifies to

$$\frac{1}{(512)^2}(2\left|\sum_{u=0}^{85}(\omega^{tr})^u\right|^2 + 4\left|\sum_{u=0}^{84}(\omega^{tr})^u\right|^2),$$

which is identical to the probability of reading t when we did make the initial partial reading of the last 5 photons.

Various quantum algorithms involve taking a partial reading similar to the one taken in Shor's factoring algorithm. Often, that reading is superfluous. However, taking it is useful for pedagogical reasons, since it simplifies notation in the rest of the argument, as was the case here.

SECTION 4: CONTINUED FRACTIONS.

We remind the reader of part of the study of continued fractions [HW, chapter 10]. For numbers g and $h \neq 0$, we let [g, h] = g + (1/h). Now for integers $a_0, a_1, a_2, \ldots a_k$, all positive except possibly a_0, we let $[a_0, a_1, a_2, \ldots, a_k] = [a_0, [a_1, [a_2, \ldots [a_{m-2}, [a_{m-1}, a_m]]\ldots.]]$.

Example: $[2, 3, 4, 5] = 2 + \dfrac{1}{[3,4,5]} = 2 + \dfrac{1}{3 + \dfrac{1}{[4,5]}} = 2 + \dfrac{1}{3 + \dfrac{1}{4 + \dfrac{1}{5}}} = 157/68.$

Exercise: Show the above process always produces fractions expressed in reduced terms.

Given a real number x, the following algorithm is well known to produce rational numbers which closely approximate x. Let $x_0 = x$ and $a_0 = \lfloor x_0 \rfloor$(the greatest integer equal to or less than x). Inductively, let $x_{i+1} = 1/(x_i - a_i)$ and $a_{i+1} = \lfloor x_{i+1} \rfloor$. (This stops if ever $a_i = x_i$.) The rational numbers a_0, $[a_0, a_1]$, $[a_0, a_1, a_2]$, $[a_0, a_1, a_2, a_3]$, etc, are known to be successively better approximations to x, and are (vaguely speaking) the best approximations possible using fractions with small denominators. These approximations are called the convergents to x.

Example: Let $x = 427/512 = x_0$. We get $a_0 = 0$, $x_1 = 1/(427/512 - 0) = 512/427$, $a_1 = \lfloor 512/427 \rfloor = 1$, $x_2 = 1/(512/427 - 1) = 427/85$, $a_2 = \lfloor 427/85 \rfloor = 5$, $x_3 = 1/(427/85 - 5) = 85/2$, $a_3 = \lfloor 85/2 \rfloor = 42$, $x_4 = 1/(85/2 - 42) = 2$, $a_4 = \lfloor 2 \rfloor = 2$, and the process stops since $a_4 = x_4$. We get the fractions 0, [0, 1] = 1, [0, 1, 5] = 5/6, [0, 1, 5, 42] = 211/253, and [0, 1, 5, 42, 2] = 427/512. Thus, the convergents to 427/512 are 0, 1, 5/6, 211/512, and finally, 427/512 itself.

STRATEGY: If the computer gives t, find the last convergent to $t/2^m$ having denominator less than n/2. Suppose it is p/q (which will be in reduced terms). Guess that r = q. (Soon, we will see that within a reasonable number of tries, you are very likely to find a case in which r really does equal q).

Definition: The last convergent to $t/2^m$ having denominator less than n/2 will be called the 'appropriate' convergent to $t/2^m$.

We remind the reader of a result of Legendre [HW, section 10.15, theorem 184].

(4.1) Theorem: If c and d > 0 are integers with $|x - c/d| < 1/(2d^2)$, then c/d is one of the convergents of x.

(4.2) Corollary: Let t be an integer with $|t - k(2^m/r)|n^2 < 2^{m+1}$. Then k/r is the appropriate convergent to $t/2^m$.

Proof: The hypothesis shows $|t/2^m - k/r| < 2/n^2 = 1/(2(n/2)^2) < 1/(2r^2)$ (using r < n/2), and so (4.1) shows k/r is a convergent to $t/2^m$. We must show it is the last convergent having denominator less than n/2. Suppose there is a later convergent, c/d, to $t/2^m$, with d < n/2. The theory of continued fractions shows $|t/2^m - c/d| < |t/2^m - k/r| < 2/n^2$. Therefore, $|c/d - k/r| < 2/n^2 + 2/n^2 = 4/n^2 = 1/(n/2)^2$. Now $|c/d - k/r| = |rc - kd|/rd$. However, d and r are both less then n/2, and so $|c/d - k/r| > |rc - kd|/(n/2)^2$. We now have $1/(n/2)^2 > |c/d - k/r| > |rc - kd|/(n/2)^2$, and since the numerator of this last fraction is an integer, it must be 0, which implies c/d = k/r, contradicting that c/d is a later convergent.

(4.3) Corollary: a) If $n^2/4 < 2^m$, then k/r is the appropriate convergent to $t_k/2^m$.

b) If $n^2 < 2^m$, then k/r is the appropriate convergent to each of $(t_k - 1)/2^m$, $t_k/2^m$, and $(t_k + 1)/2^m$.

Proof: Suppose $2^m > n^2D$. Then $|t - k(2^m/r)|n^2 < |t - k(2^m/r)|2^m/D$. If that last is at most 2^{m+1}, then (4.2) shows k/r is the appropriate convergent to $t/2^m$. Thus, if $|t - k(2^m/r)| \le 2D$, then k/r is the appropriate convergent to $t/2^m$. If D = 1/4, we need $|t - k(2^m/r)| \le 1/2$. That is true when $t = t_k$, proving part (a). If D = 1, we need $|t - k(2^m/r)| \le 2$, proving part (b). (Remark: When D = 1, one of $(t_k \pm 2)/2^m$ also has k/r as its appropriate convergent, but we will ignore that slight improvement.)

Remark: If D < 1/4, we do not know that k/r is an appropriate convergent to any $t/2^m$. That is why we use $n^2/4 < 2^m$, to be certain there is a nonzero chance of success. Letting D get larger than 1 would increase the odds of success a little, but as m increases, more photons are involved, and the process slows down. Not knowing the exact workings of the computer, I am not qualified to estimate the optimal D, but guess it is between 1/4 and 1. If I had to guess, I would say ½. The next exercise discusses that case.

(4.4) Exercise: Suppose $n^2/2 < 2^m$. Show that if $\delta_k \ge 0$, then $(t_k - 1)/2$ and $t_k/2^m$ have k/r as their appropriate convergent, while if $\delta_k \le 0$, then $t_k/2^m$ and $(t_k + 1)/2^m$ have k/r as their appropriate convergent.

Example: We complete the example developed in section 3. We saw the probability that the computer gives us one of 426, 427, or 428 is almost .15. Suppose we got 427 (probability ≈ .113). We previously saw that the list of convergents to 427/512 is 0, 1, 5/6, 211/512, 427/512. Since n/2 = 21/2 = 10.5, the appropriate convergent is 5/6. We guess r = 6, which is correct.

If instead of 427, the computer had given us 426 or 428, we would still be led to 5/6, as (4.3)(b) shows (since we are using the $n^2 < 2^m < 2n^2$ option in this example).

However, suppose the computer gives us $t_4 = 341$ (probability $\approx .113$). The last convergent to 341/512 having denominator less than 10.5 is 2/3. That is the reduced term form of 4/6. Thus, we want 6 but have 3. This problem arises when GCD(k, r) $\neq 1$. We next address that concern.

Notation: Let P be .4 if $n^2/4 < 2^m < n^2/2$, .8 if $n^2/2 < 2^m < n^2$, and .85 if $n^2 < 2^m < 2n^2$.

Proposition: The probability that we successfully factor n on the first attempt is at least $P\phi(r)/2r$, and that exceeds $P/(2(\log n))$.

Proof: In (3.3), we saw that $P^{\#}(k)$, the probability the computer gives us one of $t_k - 1$, t_k, or $t_k + 1$, is at least .85/r. Suppose $n^2 < 2^m < 2n^2$. Then P = .85, and so that probability is at least P/r. By (4.3)(b), k/r is the appropriate convergent to each of $(t_k - 1)/2^m$, $t_k/2^m$, and $(t_k + 1)/2^m$. If also, GCD(k, r) = 1, then k/r is in reduced terms, and so r is the denominator of the appropriate convergent, so that the above strategy for finding the correct r succeeds in such a case. There are $\phi(r)$ choices for such a k, and so the probability of successfully finding r is at least $P\phi(r)/r$. Recall from section 1 that if b is useful, then knowing r does lead to a successful factorization of n. Also, the probability of b being useful is at least ½. Therefore, the overall probability of success is at least $P\phi(r)/2r$. The argument in the case $n^2/4 < 2^m < n^2/2$ is similar, using P(k) instead of $P^{\#}(k)$, and (4.3)(a) instead of (4.3)(b). The case $n^2/2 < 2^m < n^2$ is also similar, using exercises (3.4) and (4.4). (Assuming $2n^2 < 2^m < 4n^2$ would not increase the odds of success very much. However, it would require more photons, and slow the process down.)

It remains to show that $\phi(r)/r > 1/(\log n)$. As r is the order of an element in the group of units modulo n, r divides $\phi(n)$. Using that, the formula for the ϕ function easily shows $\phi(r)/r \geq \phi(\phi(n))/\phi(n)$. We will show that last exceeds $1/(\log n)$. Now $\phi(n)$ is big (being bigger than $\pi(n)$, which is known to be big by the prime number theorem). We are thus justified in saying $\phi(\phi(n)) > \pi(\phi(n)) \approx \phi(n)/(\log \phi(n))$, so that $\phi(\phi(n))/\phi(n)$ can be estimated to be at least $1/(\log \phi(n)) > 1/(\log n)$. Therefore, we estimate that $\phi(r)/r > 1/(\log n)$. (That is a crude estimate. It is known that there is a constant c with $\phi(r)/r > c/(\log \log r) \geq c/(\log \log(n/2)$ [HW, theorem 328]).

There are two ways we can fail to find r. Consider the convergent c/d we get. If it does not equal any k/r, we have failed. If it does equal some k/r, but GCD(k, r) $\neq 1$, we have failed. However, in this last type of failure, we will have d divides r. It has been shown that there is some merit to taking whatever d you get, and trying small multiples of it, hoping to hit upon r. Another scheme with merit is to do the entire process again, starting with the same b. Then take the new convergent c'/d' you find. As there is a decent chance both d and d' divide r, try small multiples of LCM(d, d'). Of course, one can always start all over again, with a new random b.

Remark: Our method assumes n is not a power of a prime, since the exponential factorization method fails for powers of primes. Apparently, there exists a classical method for factoring a

power of a prime. Nonetheless, it is interesting to note that Shor's method for finding the order of b mod n (which works for any n), also allows factoring powers of primes, as we now outline.

Let $n = p^e$, p an odd prime, $e > 1$. Pick a random b with $1 < b < n$. If GCD(b, n) ≠ 1, we are done. Suppose b is a unit mod n. Using Shor's method, find r, the order of b mod n. We know r divides $p^{e-1}(p - 1)$. Using that n has a primitive root, it is not hard to see that there is a high probability that p divides r, and if so, then GCD(r, n) is a proper factor of n.

Section 5: SPEED.

Finally, we briefly (and admittedly, somewhat vaguely) discuss the time required to do all of this, which is to say, we estimate the number of separate calculations involved. Our goal is to show that it is O(poly(log n)), where 'poly' denotes some polynomial. We begin by noting that since $n^2/4 < 2^m < 2n^2$, we have O(m) = O(log n). From what I have read, it appears that operations such as multiplying two numbers of the order of magnitude of n requires $O((\log n)^2)$ calculations, and so we will not worry about details like that.

In actual operation, the steps described in sections 3 and 4 are done first, and make use of the quantum computer. In section 4, we discussed how to use the t the quantum computer produces to find (a possible) r. In section 1, we described how to use r to (hopefully) factor n. Those last two steps could be done on a classical computer, if that is judged more efficient, or cheaper.

Not knowing the physical workings of the quantum computer, I merely report what I have read of them, freely admitting that I might have some of the details wrong. (Various sources make slightly conflicting, or confusing statements.) The Hadamard gate is quick and easy. Each application is a one step process. We repeat that step m ≈ log n times.

We next used the gate S, which converted $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} |\, j,0 >$ to $(1/\sqrt{2^m}) \sum_{j=0}^{2^m-1} |\, j,b^j \bmod n >$.

Apparently that is the slowest part of the procedure, taking $O((\log n)^3)$ steps, by using fast exponentiation, described below.

The QF gate performed a Fourier transform of a sequence of length 2^m. Being a power of 2, it can be done in as a series of m(m + 1)/2 easy subgates, and so requires $O((\log n)^2)$ substeps. (Fourier transforms on sequences of any length can be accomplished, but usually take many more steps. Powers of 2 are a special, quick, case. That is why we use 2^m.)

We now turn to the continued fraction process. If one computes the convergents to some number x, and lists them as c_0/d_0, c_1/d_1, c_2/d_2, …, it can be proven (via a slightly awkward induction) that $c_{i+2} \geq 2c_i$ and $d_{i+2} \geq 2d_i$. Thus $d_{2i} \geq (2^i)d_0 \geq 2^i$. We stopped at the largest $d_i < n/2 = \sqrt{n^2/4} < \sqrt{2^m} = 2^{m/2}$. Thus, the number of convergents we need to find is O(log n). Of course, finding each convergent required various calculations (multiplications, additions, etc), but we already dealt with such.

We take the r given us by the continued fraction process, and apply the method of section 1 to it. That involves raising b to the power r/2. That can be done quickly, via 'fast exponentiation'. We illustrate with an example. To find b^{21}, write 21 = 16 + 4 + 1 =

$2^4 + 2^2 + 1$. Calculate $b^2 = c$, $(b^2)^2 = d$, $((b^2)^2)^2 = e$, and $(((b^2)^2)^2)^2 = f$. Note that $b^{21} = (b)(d)(f)$. Apparently, when the various multiplications are included, the process takes O(poly(log n)) steps.

Concerning the Euclidean algorithm used to find GCD($b^r/2 - 1$, n), an induction (a bit easier than the one for continued fractions) shows that the remainders which appear in that algorithm get cut in half (or more) at every two repetitions of the algorithm, and so there are O(log n) repetitions of the algorithm. (Again, the multiplications, additions, etc, have been dealt with.)

The above shows that one attempt at factoring n takes O(poly(log n)) steps. In the previous section, we saw that a single attempt has probability at least P/2(log n) of being successful, and so O(log n) attempts will assure success to an arbitrarily high probability.

SECTION 6: DISCRETE LOGS.

Let p be a large prime. It is known that the group of units modulo p is cyclic. Let g be a generator. Given an integer r with $0 \leq r \leq p - 1$, one can quickly find $g^r \bmod p - 1$. However, given $g^r \bmod p - 1$, there is no quick classical way of finding r. That is called the discrete log problem. An encryption system (the El Gamal system) is based upon that fact. We present (a slight variation on) Shor's method for using a quantum computer to find r quickly. We also show that the method is more efficient than [S] indicates, by analyzing the technique more closely.

Notation: r will be the exponent of g we seek, (Note that $0 \leq r < p - 1$, and so $r = r \bmod (p - 1)$). $q = 2^m$ will be a power of 2 with $4p < q < 8p$. We will write $p - 1 = 2^k D$ with D odd. The integers c and d will be between 0 and q – 1 (inclusive). We can uniquely define integers K(c) and R(c) by writing $c(p - 1) = K(c)q + R(c)$, with $-q/2 < R(c) \leq q/2$. We have $R(c) \equiv c(p - 1) \bmod q$. Let D(c) = GCD(K(c), p – 1). Let $T(c, d) = d + r[c - R(c)/(p - 1)]$. Let u(d) be the integer such that $u(d)/(p - 1)$ is the integral multiple of $1/(p - 1)$ closest to $-d/q$. (If $-d/q$ happens to lie halfway between two integral multiples of $1/(p - 1)$, take either choice for u(d). However, that will not happen in the cases that interest us.)

The computer will be programmed in such a way that the answer we read from it will be a pair of integers (c, d), with $0 \leq c,d \leq q - 1$.

(6.1) Lemma: D(c) and 2^k both divide R(c).

Proof: Since D(c) = GCD(K(c), p – 1), and 2^k = GCD(p – 1, q), the result follows from $c(p - 1) = K(c)q + R(c)$.

Definition: The pair (c, d) is alluring if $jq - 1/2 < T(c, d) \leq jq + 1/2$ for some integer j.

(6.2) Lemma: For c (with $0 \leq c \leq q - 1$), there exists a unique d (with $0 \leq d \leq q - 1$) such that (c, d) is alluring.

Proof: This follows easily from the fact that T(c, d) increases by 1 each time d increases by 1, and there are q consecutive choices for d.

Remarks: a) Since we do not know r, we cannot directly tell if the (c, d) read from the computer is alluring.

b) In general, the definition of u(d) shows that u(d)/(p – 1) is within 1/(2(p – 1)) of –d/q. However, if (c, d) is alluring, the next proof shows u(d)/(p – 1) is within 1/(2q) of –d/q, a better approximation, since q > 4p.

(6.3) Lemma: Suppose (c, d) is alluring. Then $rK(c) \equiv u(d) \bmod (p-1)$. If also $D(c) = GCD(K(c), p-1)$ is 1, (so that K(c) has an inverse mod (p – 1)), then $r \equiv (K(c))^{-1}u(d) \bmod (p-1)$. Finally, if c and d are known (say being read from the computer), then r can be found efficiently (which is the goal).

Proof: We know $jq - 1/2 < T(c, d) \leq jq + 1/2$ for some integer j. Dividing by q shows

$j - 1/(2q) < T(c, d)/q \leq j + 1(2q)$. Now $T(c, d)/q = (1/q)[d + r(c – (R(c)/(p – 1)))] =$

$(1/q)[d + r(c(p – 1) – R(c))/(p – 1)] = (1/q)[d + rK(c)q/(p – 1)] = d/q + rK(c)/(p – 1) =$

$rK(c)/(p – 1) – (-d/q)$. Thus, we have $j – 1/(2q) < rK(c)/(p – 1) – (-d/q) \leq j + 1/(2q)$, so that

$-1/(2q) < rK(c)/(p – 1) – (-d/q) – j \leq 1/(2q)$.

Thus, $|[rK(c) – j(p – 1)]/(p – 1) – (-d/q)| \leq 1/(2q)$.

Since $4p < q$, we have $1/q < 1/4p < 1/(p – 1)$. Therefore, at most one integral multiple of $1/(p – 1)$ lies within distance $1/(2q)$ of –d/q. In our case there is one, namely $[rK(c) – j(p – 1)]/(p – 1)$, which therefore must be $u(d)/(p – 1)$, showing $u(d) = rK(c) – j(p – 1)$. Since we do not know what j equals, we can only say $rK(c) \equiv u(d) \bmod (p-1)$. However, if $D(c) = 1$, then $r \equiv (K(c))^{-1}u(d) \bmod (p-1)$. Obviously if c and d are known, then K(c), u(d), and D(c) can be found efficiently, so that if D(c) = 1, then r can be found efficiently.

STRATEGY: Make successive computer runs. For each (c, d) produced, if $D(c) \neq 1$, go to the next run. If D(c) = 1, then calculate $(K(c))^{-1}u(d) \bmod (p-1)$, and test whether is the sought after r (i.e., whether g raised to that number equals $g^r \bmod p$. (Note that (6.3) shows that if (c, d) is alluring, the candidate will actually be r.)

Remarks: An alternate strategy would be to not restrict ourselves to cases in which D(c) = 1, but to also work with small D(c), (where "small" is negotiable). From $rK(c) \equiv u(d) \bmod (p-1)$, we get $rK(c)/D(c) \equiv u(d)/D(c) \bmod (p-1)/D(c)$. As K(c)/D(c) is relatively prime to (p – 1)/D(c), that lets us find r mod (p – 1)/D(c). Since D(c) is small, a short search then leads to r.

Shor's original approach was to allow any D(c) and argue that knowing r mod (p – 1)/D(c) for a lot of D(c) allows finding r mod (p – 1)/G where G is the (hopefully small) GCD of all the D(c). However, that approach is complicated by the fact that one cannot know in advance that all the D(c) being used come from alluring (c, d).

Note that $rK(c) \equiv u(d) \bmod (p-1)$ implies D(c) divides u(d). Thus, if D(c) does not divide u(d), then (6.3) shows (c, d) is not alluring. However, our strategy sidesteps having to worry about whether or not (c, d) is alluring.

In order to show the strategy has a good chance of finding r within a not too large number of computer runs, we must show the probability of finding an alluring (c, d) with D(c) = 1 is acceptably high. If we just chose a (c, d) at random, the odds of it being alluring are 1/q (using (6.2) and the fact that there are q^2 choices for (c, d)). That probability is much too small for our purposes. However, the computer will be programmed to make the appearance of an alluring (c, d) be much higher than 1/q. To explain how, we need to focus on a subset of the alluring (c, d).

Definition: c is good if $|R(c)| \le q/16$. Also, (c, d) is good if it is alluring and c is good.

PREVIEW: The computer will be biased so as to produce any *specified* good (c, d) with probability greater than 1/16q (as opposed to $1/q^2$). We will also see there are a fairly high number of good (c, d) with D(c) = 1. Combining those two facts shows there is a fairly high chance of getting a good (hence alluring) (c, d) with D(c) = 1, meaning r can be found via (6.3). (Allure is used to make (6.3) work. Goodness is used to assure adequately high odds of success.)

We justify our preview at the end. For now, we accept it, and use it to show that the probability of getting a good (c, d) with D(c) = 1 is adequately high.

(6.4) Lemma: a) For $-q/2 < R \le q/2$, R = R(c) for some c ($0 \le c < q$) exactly when 2^k divides R.

b) R(c) = R(c') IFF $c \equiv c' \bmod 2^{m-k}$. There are 2^{m-k} distinct R(c), and each occurs 2^k times (as c varies from 0 to q – 1). Furthermore, if $c' = c + 2^{m-k}h$ then K(c') = K(c) + hD.

c) If R = R(c) for some c, then there are 2^{k-1} choices of c for which R(c) = R and D(c) is odd.

Proof: a) Suppose R = R(c). By (6.1), 2^k divides R. Conversely, suppose 2^k divides R. Since $GCD((p-1)/2^k, q/2^k) = 1$, there is an integer c with $c(p-1)/2^k \equiv R/2^k \bmod q/2^k$ and with $0 \le c < q/2^k < q$. Now $c(p-1) \equiv R \bmod q$. Since $-q/2 < R \le q/2$, the definitions show R = R(c).

b) Suppose $c' = c + h2^{m-k}$. Then $c'(p-1) = c(p-1) + h2^{m-k}(p-1) = R(c) + K(c)q + h2^{m-k}(2^kD) = R(c) + K(c)q + hqD$. Since $-q/2 < R(c) \le q/2$, the definition of $R(c')$ shows it equals R(c), and the definition of $K(c')$ shows it equals K(c) + hD. Conversely, if $R(c') = R(c)$, then $c'(p-1) = R(c') + K(c')q = R(c) + K(c')q = [c(p-1) - K(c)q] + K(c')q$, so that $(c' - c)(p-1) = q(K(c') - K(c))$. Dividing by 2^k gives $(c' - c)D = 2^{m-k}(K(c') - K(c))$. Since D is odd, $c' \equiv c \bmod 2^{m-k}$. The rest of the result follows.

c) Suppose R = R(c), with c the smallest such. By (b), the set of c' with $R(c') = R$ is $\{c' = c + h2^{m-k} \mid 0 \le h < 2^k\}$. From (b), we have $K(c') = K(c) + hD$. Now D is odd, and half of the h are odd. Thus, for exactly half of the c', $K(c')$ is odd, and so for exactly half the c', $D(c') = GCD(K(c'), p-1)$ is odd. By (b), there are 2^k c' in total, and so half is 2^{k-1}.

Notation: Let $S = \{s \mid -2^{m-k-4} \le s \le 2^{m-k-4}$ and $GCD(s, D) = 1\}$.

(6.5) Theorem: The probability that the above strategy will successfully find r on any given run is greater than $|S|/2^{m-k+5}$.

Proof: Fix an s in S, and let $R = 2^ks$. Now $|R| = 2^k|s| \le 2^k2^{m-k-4} = q/16$. By (6.4)(a)(c), there are 2^{k-1} choices of c with R(c) = R and D(c) odd. As $|R(c)| \le q/16$, all those c are good. By (6.2), for each of those c, there is a alluring (hence good) (c, d). Also, (6.1) shows D(c) divides $R = 2^ks$, and since D(c) is odd, it divides s. However, by definition, D(c) also divides $p - 1 = 2^kD$, and so D(c) divides GCD(s, D), showing D(c) = 1. Therefore, the above strategy would find (c, d) an acceptable reading, and it would lead to finding r.

We have just seen that any s in S gives rise to 2^{k-1} good (c, d) for which D(c) = 1. Now no (c, d) arises from two different choices of s, since if c arises from s, then $R(c) = 2^ks$. Thus the total number of good (c, d) under consideration is $2^{k-1}|S|$. By the preview, each has probability greater than $1/16q = 1/2^{m+4}$ of being read. The result follows.

(6.6) Lemma: $2^{m-k-3} \le D < 2^{m-k-2}$ and $k \le m - 3$. Also, D = 1 exactly when k = m - 3.

Proof: Since $4p < 2^m < 8p$, we have $2^{m-3} \le p - 1 < p < 2^{m-2}$. Using $p - 1 = 2^kD$ shows $2^{m-k-3} \le D < 2^{m-k-2}$. Since $D \ge 1$, that shows $k \le m - 3$. The two constraints on D also show D = 1 exactly when k = m – 3. (Of course, D = 1 exactly when p is a Fermat prime, of which, only five are known.)

(6.7) Corollary: a) If k = m – 3, the probability the above strategy successfully finds r on any given run is greater than $1/2^8$.

b) If $m - 12 \le k \le m - 4$, the probability the above strategy successfully finds r on any given run is greater than $(m - k - 3)/2^{m-k+4} \ge 9/2^{16}$.

c) If $k \le m - 13$, the probability the above strategy successfully finds r on any given run is greater than $1/(2^{10}(m - k - 4)) > 1/(2^{10}m)$. (Note that since $4p < 2^m < 8p$, m is slightly more than log p/log2.)

Proof: a) When $k = m - 3$, (6.6) shows $D = 1$, and so in this very special case, we see $S = \{0\}$. Use (6.5).

b) Since D is odd, S clearly contains all the powers of 2 between 1 and 2^{m-k-4}. There are $m - k - 3$ of them, and their negatives are also in S, showing $|S| \ge 2(m - k - 3)$. By (6.5), the probability of finding r on any given run is greater than $(m - k - 3)/2^{m-k+4}$. For the values of k in (b), that quantity is at least $9/2^{16}$.

c) By (6.5), it will suffice to show $|S| \ge \dfrac{2^{m-k-5}}{m-k-4}$. Since s is in S if and only if –s is in S, it will suffice to show S contains at least $\dfrac{2^{m-k-6}}{m-k-4}$ positive numbers. We will show the primes in S together with the powers of 2 in S contribute at least that number of elements to S.

We start by considering primes less then 2^{m-k-4}. (All of them will be in S except those that divide D.) A result of Chebyshev [F, Chapter 1, Theorem 5.5] shows that the number of primes less than $x \ge 2$, exceeds $\dfrac{x \log 2}{4 \log x}$. Thus the number of primes less that 2^{m-k-4} exceeds $\dfrac{2^{m-k-6}}{m-k-4}$. Suppose that t of those primes divide D. We claim $t \le m - k - 4$. For the range of k in (c), we have $m - k - 4 \ge 9$, and so if $t \le 9$ we are done. Suppose $t > 9$. Now D is at least as large as the product of those t (distinct) primes, and so using (6.6), $2^{m-k-2} > D > (3)(5)(7)(11)^{t-3}$. Taking logs, we see $m - k - 2 > (m - k - 2)\log 2 > \log(3) + \log(5) + \log(7) + (t - 3)\log 11 > 4 + 2(t - 3)$. Thus $t < (m - k)/2 \le m - k - 4$ (for our k).

The preceding shows that S contains at least $\dfrac{2^{m-k-6}}{m-k-4}$ - $(m - k - 4)$ primes. However, as in the proof of (b), S also contains $m - k - 3$ powers of 2. Ignoring the number 2, (which we just treated as a prime), the other powers of 2 contribute $m - k - 4$ new elements to S, and we are done. (The bound in (c) also works for some larger k, but for those k the bound in (b) is better.)

It only remains to justify the probability given in the preview. For that, we must discuss the computer program. Recall $4p < q = 2^m < 8p$. The computer will have three registers. The first two registers will both deal with numbers from 0 to $q - 1 = 2^m - 1$, and so will require m qubits each. The third register will deal with numbers mod p, and so will need $m - 2$ qubits, since $2^{m-2} > p$. In all, $3m - 2$ qubits are needed. Recall that we took g to be a cyclic generator for the group of units modulo p, we are assuming we know g^r mod p, and we seek r.

The computer is initially put into the state $(1/(p - 1)) \sum_{a,b=0}^{p-2} |a,b,0>$. As discussed near the end of section 2, we can have the computer convert that into the state

$(1/(p-1))\ \sum_{a,b=0}^{p-2} |a,b,g^a (g^r)^{-b} \bmod p>$.

With $\omega = e^{2\pi i/q}$, the set of vectors $\{(1/q)\sum_{c,d=0}^{q-1} \omega^{ac+bd} |c,d> \mid 0 \le a \le q-1,\ 0 \le b \le q-1\}$ is an orthonormal basis of 2m-qubit space, and so there is a quantum gate taking |a, b> to $(1/q)\ \sum_{c,d=0}^{q-1} \omega^{ac+bd} |c,d>$. Applying that gate to the first 2m qubits of our previous state converts it to $\frac{1}{(p-1)q} \sum_{a,b=0}^{p-2}\ \sum_{c,d=0}^{q-1} \omega^{ac+bd}$ |c, d, $g^{a-rb} \bmod p$>.

If the computer is now read, the reading will have the form (c, d, $g^s \bmod p$) for some s with $0 \le s < p-1$. (In what follows, s is constant.) The probability of getting that reading is the square of the norm of the coefficient of |c, d, g^s> in the above state. That coefficient is easily seen to be $\frac{1}{(p-1)q} \sum \omega^h$, the summation over all h = ac + bd with $a - rb \equiv s \bmod (p-1)$ (and a and b ranging from 0 to p – 2). Thus, the probability of reading (c, d, $g^s \bmod p$) is $(\frac{1}{(p-1)q})^2 N^2$, where $N = |\sum \omega^h|$. We claim that if (c, d) is good, then $N > (p-1)/\sqrt{2}$.

For each b with $0 \le b \le p-2$, there is a unique a ($0 \le a \le p-2$) such that $a - rb \equiv s \bmod (p-1)$, and an integer m_b with $a = rb + s - m_b(p-1)$.
Now h = ac +bd = rbc + sc - $m_b(p-1)c$ + bd, and so $N = |\sum_{b=0}^{p-2} \omega^{rbc+sc-m_b(p-1)c+bd}|$.
We can factor ω^{sc} out of each term, and as that factor has norm 1, we may ignore it.
Therefore, $N = |\sum_{b=0}^{p-2} \omega^{rbc-m_b(p-1)c+bd}|$.

Recall that T(c, d) = d + r[c – R(c)/(p – 1)]. Let V(b, c) = [(rb/(p – 1)) – m_b]R(c). We then see that modulo q, we have rbc – m_b(p – 1)c + bd = rbc + bd - m_b(K(c)q + R(c)) ≡ rbc + bd - m_bR(c) = bT(c, d) + V(b, c) mod q. Since $\omega^q = 1$, we see that $N = |\sum_{b=0}^{p-2} \omega^{bT(c,d)} \omega^{V(b,c)}|$. (We have not yet used that (c, d) is good.)

Now consider a *specific* good (c, d). Then c is good, so that $|R(c)| \le q/16$. Also, since we have a = rb + s – m_b(p – 1), we have rb/(p – 1) – m_b = (a – s)/(p – 1). Since a and s are both between 0 and p – 2, we see |rb/(p – 1) – m(b)| < 1. It immediately follows that |V(b, c)| < q/16.

We now know that $N = |\sum_{b=0}^{p-2} \omega^{bT(c,d)} \omega^{V(b,c)}|$ and |V(b, c)| < q/16. We cannot similarly bound |bT(c, d)|, so we instead replace it with something congruent to it mod q. We recall that (c, d) being good implies it is alluring, and so $-1/2 < T(c, d) - jq \le 1/2$ for some integer j. Write T(c, d) = jq + ε, with $|\varepsilon| \le ½$. Then $\omega^{bT(c,d)} = \omega^{bjq+b\varepsilon} = \omega^{b\varepsilon}$. Therefore, $N = |\sum_{b=0}^{p-2} \omega^{b\varepsilon} \omega^{V(b,c)}|$. Now $|b\varepsilon| = b|\varepsilon| \le (p-2)|\varepsilon| < p(1/2) = p/2$. However, we want something a bit stronger, which we get as follows. Since $|\omega^{-(p-2)\varepsilon/2}| = 1$, we see $N = |\sum_{b=0}^{p-2} \omega^{(b-(p-2)/2)\varepsilon} \omega^{V(b,c)}|$. For $0 \le b \le p-2$, we have $|(b - (p-2)/2)\varepsilon| \le ((p-2)/2)(1/2) < p/4 < (q/4)/4 = q/16$.

We now see that $|(b - (p-2)/2)\varepsilon + V(b, c)| < q/16 + q/16 = q/8$.

As $\omega = e^{2\pi i/q}$, N is the norm of a sum of terms each having form $e^{\theta i}$, where for each term, $|\theta| < (2\pi/q)(q/8) = \pi/4$. Therefore, the real part of each term in that sum is greater than $\cos(\pi/4) = 1/\sqrt{2}$. As there are p – 1 terms in that sum, its real part is greater than $(p-1)/\sqrt{2}$. Now the norm of that sum is at least as large as its real part, and so $N > (p-1)/\sqrt{2}$, as claimed.

We now see the probability of the computer reading giving (c, d, g^s mod p) for our specified good (c, d) and some fixed s ($0 \le s < p-1$) is $(\frac{1}{(p-1)q})^2 N^2 > (\frac{1}{(p-1)q})^2 (\frac{p-1}{\sqrt{2}})^2 = 1/(2q^2)$.

As there are p – 1 possible s (and we do not care which we have), the probability of getting some (c, d, g^s mod p) for our specified good (c, d) is greater than $(p-1)/(2q^2)$.

As $q < 8p$, (and as $q/8 = 2^{m-3}$ is an integer), we have $q/8 \le p-1$, so that $(p-1)/q \ge 1/8$, showing the probability of getting our specified good (c, d) is greater than 1/(16q), as in the preview.

Remark: Shor assumed $p < q < 2p$, and defined c to be good if $|R(c)| \le q/12$. We modified both of those, so as to have $\pi/4$ appear in the previous proof, which is convenient. Also, a few other arguments are simplified. However, it requires a few extra qubits, and so would slow computer runs down slightly.

FURTHER READING

[F] Daniel E. Flath, Introduction to number theory, Wiley-Interscience, New York, 1989.

[G] Stan Gudder, Quantum Computation, MAA Monthly, v. 110, number 3, March 2003, pp. 181-201].)

[HW] G. H.Hardy and E. M. Wright, An introduction to the theory of numbers (5-th edition), Oxford University Press, New York, 1979.

[J] Richard Jozsa, Quantum factoring, discrete logarithms, and the hidden subgroup problem, available at xxx.lanl.gov/abs/quant-ph/0012084

[M1] Stephen McAdam, Entanglement, available at www.ma.utexas.edu/users/mcadam

[M2] Stephen McAdam, Bell's theorem and the demise of local reality, Amer. Math. Monthly, 110 (2003) 800-811.

[RP] Eleanor Rieffel and Wolfgang Polak, An introduction to quantum computing for non-physicists, available at xxx.lanl.gov/abs/quant-ph/9809016

[S] Peter Shor, Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer, available at xxx.lanl.gov/abs/quant-ph/9508027