then the row reduction of the matrix would all be very time consuming. Conversely, if we choose y fairly small, then the latter tasks would be easy, but it would take us a very long time to find any b_i's for which $b_i^2 \bmod n$ is divisible only by primes $\leq y$, because in that case u^u would be very large. So y should be chosen in some intermediate range, as a compromise between these two extremes.

In order to decide how y should be chosen, we first make a very rough estimate in terms of y (and n, of course) of the number of bit operations. We then minimize this with respect to y (using first year calculus and some simplifying approximations), and find our time estimate with y chosen so that the time is minimized.

Suppose that n is an r-bit integer and y is an s-bit integer; then u is very close to r/s. First of all, how many bit operations are needed for each test of a randomly chosen b_i? We claim that the number of operations is polynomial in r and y, i.e., it is $O(r^l e^{ks})$ for some (fairly small) integers k and l. It takes a fixed amount of time to generate a random bit, and so $O(r)$ bit operations to generate a random integer b_i between 1 and n. Next, computing $b_i^2 \bmod n$ takes $O(r^2)$ bit operations. We must then divide $b_i^2 \bmod n$ successively by all primes $\leq y$ which divide it evenly (and by any power of the prime that divides it evenly), hoping that when we're done we'll be left with 1. A simple way to do this (though not the most efficient) would be to divide successively by 2 and by all odd integers p from 3 to y, recording as we go along what power of p divides $b_i^2 \bmod n$ evenly. Notice that if p is not prime, then it will not divide evenly, since we will have already removed from $b_i^2 \bmod n$ all of the factors of p. Since a division of an integer of $\leq r$ bits by an integer of $\leq s$ bits takes time $O(rs)$, we see that each test of a randomly chosen b_i takes $O(rsy)$ bit operations.

To complete step (1) requires testing approximately $u^u(\pi(y)+1)$ values of b_i, in order to find $\pi(y)+1$ values for which $b_i^2 \bmod n$ is a product of primes $\leq y$. Since $\pi(y) \approx \frac{y}{\log y} = O(y/s)$, this means that step (1) takes $O(u^u r y^2)$ bit operations.

Step (2) then involves operations which are polynomial in y and r (such as matrix reduction and finding b and c modulo n). Thus, step (2) takes $O(y^j r^h)$ bit operations for some integers j and h. Each time we perform steps (1)–(2) there is at least a 50% chance of success, i.e., of finding that $b \not\equiv \pm c \bmod n$. More precisely, the chance of success is 50% if n is divisible by only two distinct primes, and is greater if n is divisible by more primes. Thus, if we are satisfied with, say, a $1-2^{-50}$ probability of finding a nontrivial factor of n, it suffices to go through the steps 50 times. Taking this as good enough for all practical purposes, we end up with the estimate

$$O(50(u^u r^2 y^2 + y^j r^h)) = O(r^h u^u y^j) = O(r^h u^u e^{ks}) = O(r^h (r/s)^{r/s} e^{ks}),$$

for suitable integers h and k.

We now find y — equivalently, s — for which this time estimate is minimal. Since r, the number of bits in n, is fixed, this means minimizing