IMPROVING TRIAL DIVISION: FIRST-DIGIT ANALYSIS

FRANCO CAFIERO

The purpose of the research thus far has been to improve the trial-division algorithm for factoring integers. Trial division factorization involves dividing the target number (call it n) by all primes less than or equal to $\sqrt{n}$ until an integral quotient (and hence a prime factor) is found. Clearly this algorithm can become very time consuming for large integers. However, given a sufficiently intelligent implementation of the algorithm over certain types of integers, it may be possible to reduce factorization time considerably.

The specific problem invesigated so far in our research has been that of factoring an integer n that is the product of two prime numbers, p and q, of equal length (i.e. p and q have the same number of digits). Composite numbers of this type appear commonly in RSA encryption; for our purposes, we will refer to such composite numbers as RSA composites. We know that since n is composite, n must have a prime factor less than or equal to $\sqrt{n}$. Assume without loss of generality that $p < q$; we know that p is bounded from above by $\sqrt{n}$, but what number bounds p from below? Finding this lower bound was the main component of the investigation. Once a sufficiently large general lower bound is found, then trial division can be used to find the prime factor p of n relatively quickly.

Let us introduce a lemma before we proceed:

Lemma 1. *Let $a, b \in \mathbb{Z}$, and let a have s digits and b have t digits. Then the product ab has either $s+t$ or $s+t-1$ digits.*

Proof. Let a and b be as above. We know 10^n has $n+1$ digits. So $a < 10^s$ and $b < 10^t$; now $ab < 10^{s+t}$. Clearly 10^{s+t} has $s+t+1$ digits, and is the smallest integer with this many digits. Since $ab < 10^{s+t}$, then ab has at most $s+t$ digits.

Now $a \geq 10^{s-1}$ and $b \geq 10^{t-1}$; now $ab \geq 10^{s+t-2}$. Clearly 10^{s+t-2} has $s+t-1$ digits, and since $ab \geq 10^{s+t-2}$, then ab must have at least $s+t-1$ digits. □

We wish to find the prime factors p and q of n. Let k denote the number of digits in n. By the lemma, we know that if k is even, then p and q must each have $\frac{k}{2}$ digits, and if k is odd, p and q must each have $\frac{k+1}{2}$ digits.

We know that $\sqrt{n}$ is the upperbound for our smaller factor p, and it makes sense; since $(\sqrt{n})^2 = n$, $\sqrt{n}$ has $\frac{k}{2}$ or $\frac{k+1}{2}$ digits. By similar reasoning, integers below $\sqrt[3]{n}$ are too small, as they have too few digits. As such, our first lower bound was set to $n^{\frac{5}{12}}$, with $\frac{1}{3} < \frac{5}{12} < \frac{1}{2}$. This lowerbound worked out reasonably well for very small numbers, but becomes time-consuming for larger numbers. This is because $n^{\frac{5}{12}}$ begins to have far fewer digits than $\sqrt{n}$ as n gets larger.

The next step, then, was to restrict further the neighborhood of possible prime factors of n; the smaller the neighborhood, the faster the trial division factorization. The problem with $n^{\frac{5}{12}}$ is that for larger numbers, integers close to $n^{\frac{5}{12}}$ have too few digits. The obvious solution, then, was to restrict the lower bound such that the possible factor neighborhood include only numbers with $\frac{k}{2}$ or $\frac{k+1}{2}$ digits. So the

lower bound was raised to $10^{\frac{k}{2}-1}$ or $10^{\frac{k+1}{2}-1}$. There were no significant decreases in running time with this new lower bound.

The best and most recent progress in refining the trial division algorithm has involved raising the lower bound based on the first two digits of the number to be factored. For example, consider four-digit primes whose first digit is 2. The largest possible such prime is 2999, while the largest four-digit prime is 9973. Their product is 29909027; since this product comes from the largest four-digit prime and the largest four-digit prime beginning with two, any RSA composite larger than 29909027 cannot possibly have a prime factor beginning with two. Hence the first digit of the smaller prime factor of RSA composites larger than 29909027 must be 3 or greater.

We extend this example to the general case, and modify it accordingly for generality. We round $i999\cdots99$ (x digits) to $(i+1)10^{x-1}$, and for the largest possible corresponding $x-$digit prime we simply round to 10^x. We take the product $((i+1)10^{x-1})(10^x) = (i+1)10^{2x-1}$, which is a $2x-$digit number that represents the non-inclusive maximal product involving $x-$digit primes beginning with i.

For example, consider once more 2999 and 9973; we round them to 3000 and 10000 and take their product, which is 30000000. Clearly no RSA composite greater than 30000000 has a four-digit prime factor beginning with 2, so we can begin our trial division at a lower bound of 3000. This lower bound remains valid until 40000000, which is the maximal product involving four-digit primes beginning with 3. We use this method to construct the theorem on which our improved trial division algorithm is based.

Theorem 2. *Let n be an RSA composite with k digits. Let f equal the first two digits of n. If f is in the range $[10a, 10a+9]$ for $1 \leq a \leq 9$, then the smaller prime factor p of n has a first digit greater than or equal to a.*

Proof. Let n be an RSA composite as above, and define f and a as above. The first two digits of n, then, are in the range $[10a, 10a+9]$. Assume by way of contradiction that there exists a prime factor p of n where the first digit of p is less than a. Let p_0 be the first digit of p; now $p_0 < a$.

To streamline our proof we define an auxilliary variable k_* to be the following:

$$k_* := \begin{cases} \frac{k}{2} & \text{if } k \text{ is even} \\ \frac{k-1}{2} & \text{if } k \text{ is odd} \end{cases}$$

Since $p|n$ and n is an RSA composite, there exists a prime number q such than $n = pq$ and q has k_* digits; so $q < 10^{k_*}$. But $pq < 10^{k_*}p$, and the first two digits of $10^{k_*}p$ are $10(p_0p_1)$ where p_1 is the second digit of p. For our purposes, as will be shown shortly, we can ignore this p_1 and approximate the first two digits of $10^{k_*}p$ as $10p_0$. Now $10p_0 < 10a$ but $pq = n$; this is impossible, as the first digit p_0 of $10^{k_*}p$ is less than the first digit a of n, which implies $pq < n$. Hence p has a first digit greater than or equal to a. □

What this theorem implies, then, is that if the first digit of an RSA composite is 2, then the smaller prime factor has a first digit 2 or greater, and so on and so forth for 3,4,...,9. For RSA composites beginning with 1, we simply begin the trial divsion at 10^{k_*}.

It is now prudent to discuss the implementation of this method. Because of its many number-theoretic functions, Maple was chosen as the program for implementation. The program (whose code is included in the appendix), accepts as input an integer assumed to be an RSA composite. The length of the square root is calculated in order to restrict trial division over integers of the same length. The first two digits are extracted from the input and an if-then statement is used to determine in what range they lie; consequently, an appropriate lower bound of possible prime factors is calculated based on the theorem. Finally, trial division is carried out over a subset of a pre-loaded set of primes; the subset is bounded from above and below, respectively, by $\sqrt{n}$ and by the appropriate lower bound from the theorem. The trial division terminates as soon as a division yields an integer quotient. The prime factor, original input number, and running time are displayed.

The first-digit analysis method runs signifanctly faster than either of the two previous modified trial division algorithms. However, as must be expected, this improved trial division algorithm does not run in polynomial time. Conventional trial division runs in exponential time, specifically $O(10^{\frac{k}{2}})$ where k is the length of n.

The range of potential divisors over which trial division is carried out is $[a \cdot 10^{k_*}, \sqrt{n}]$, where a is the first digit of n. Of course in our trials we divided only over primes in this range, and so the number of possible trial divisions is $\phi(\sqrt{n}) - \phi(a \cdot 10^{k_*})$. Clearly there are greater savings in number of calculations as $a \to 9$.

Another consideration in runtime (and indeed in implementation) of the algorithm is the use of a set of primes. We were able to compile lists of all primes up to seven digits long and use these in implementation. However, after a certain point it becomes unfeasible to use a list of primes in the algorithm; compiling the list would require great amounts of time, as would extracting the appropriate subset of primes for each n. This critical point, given our resources, is in the range of 16-digit RSA composites. For 17 or more digits the algorithm must be modified to divide over all odd numbers in the appropriate range from the theorem; the appropriate primes are a subset of the set of all such odd numbers, hence the algorithm will still run as planned, but not as efficiently. Nonetheless, the increase in possible divisors offsets to an extent the time to compile a list of primes and to extract an appropriate subset.

For small RSA composites (that is, for composites that are the product of two same-length primes), it is possible to perform a modified trial division algorithm to find the smaller (and consequently the larger) prime factor in a reasonable amount of time. The most sophisticated form of modified trial division for the RSA composites at hand involves calculating a lower bound based on the first two digits of the number to be factored. As per Theorem 1, the first digit of the number n to be factored is less than or equal to the first digit of the smaller prime factor p. While this method is immensely faster than conventional trial division and the two previous implementations described herein, the first-digit analysis method still runs in wretched exponential time and is indeed much slower than other factorization algorithms, such as Pollard's rho method and the Morrison-Brillhart algorithm.

All calculations and algorithm implementations were run in Maple 6. The computer we employed used an AMD Athlon XP processor at 1.3 GHz and 256 MB of RAM at 133 MHz.

1. APPENDIX

Maple Code for First-digit Analysis Factorization Method

```
[> fdam1:=proc(n)
[>    local p,upbnd,lwbnd,a,i,settime,cpu_time,nbhd,ftd,redn,numdigits,pwr,pwr2:
[>    Digits:=200:
[>    numdigits:=length(n):
[>    pwr:=numdigits-2:
[>    redn:=n/(10.0^pwr):
[>    ftd:=floor(redn):
[>    upbnd:=floor(sqrt(n)):
[>    pwr2:=length(upbnd)-1:
[>
[>    for i from 1 to 9 do
[>       if 10*i <= ftd and ftd <= (10*i + 9) then lwbnd:=i*(10^pwr2):
[>          break:
[>       fi:
[>    od:
[>
[>    settime:=time():
[>    nbhd:=select(x -> x>= lwbnd, smallprimes):
[>    for i in nbhd do
[>       p:=n/i:
[>       a:=floor(p):
[>       if p=a then print(cat("A prime divisor, ",i," , has been found!"));
[>          break:
[>       fi:
[>    od;
[>    cpu_time:=time() - settime:
[>    print(cat("Algorithm took ",cpu_time," seconds to complete."));
[> end;
```

TABLE 1. Table of RSA composites and corresponding factorization times for each modified trial division algorithm (in seconds)

n	$n^{\frac{5}{12}}$ **Method**	10^{k_*} **Method**	**First-digit Method**
209559868321	2.424	2.353	1.767
213344369261	3.113	2.994	2.439
223073832199	2.014	2.013	1.422
230519633177	3.014	3.054	2.438
234313231457	2.073	1.983	1.432
242283403111	1.843	1.763	1.196
254754784873	2.684	2.563	2.003
262594398823	2.433	2.394	1.823
278709243983	2.694	2.524	2.018
286534000381	2.033	2.043	1.362
294276636151	3.305	3.324	2.749
295370641871	3.635	3.525	3.114
297992242981	3.125	2.975	2.528
317967342907	3.555	3.545	2.294
326012297651	3.124	2.984	1.697
337169306233	2.614	2.644	1.327
337297399837	2.444	2.343	1.026
355294003271	2.693	2.534	1.252
368750930113	3.305	3.335	2.068
381825004963	3.936	3.805	2.613
394394310901	3.124	2.984	1.752
398437181341	3.575	3.556	2.298
411035712671	4.216	4.166	2.183
415595165387	3.085	3.004	1.006
418512124763	3.605	3.546	1.571
449685185641	4.156	4.085	2.158
473653400347	3.395	3.305	1.357
505347415123	3.905	3.806	1.157
511784812879	3.655	3.575	0.831
515058902873	4.637	4.566	1.923
532512606701	3.846	3.856	1.206
541009739839	4.575	4.486	1.963
544007941391	4.237	4.166	1.527
563489621983	4.987	4.907	2.419
573251348017	4.146	4.046	1.452
591880602569	5.037	4.957	2.333
661583146787	4.647	4.567	1.202
681683184233	5.157	5.077	1.887
716029254319	5.428	5.358	1.377
723791463877	5.048	4.897	1.001
754519748753	5.447	5.367	1.442
875608796627	5.989	5.939	1.367
922677565549	6.379	6.189	1.001