INVITED PAPER.

MODERN FACTORIZATION METHODS

HANS RIESEL

Department of Numerical Analysis and Computing Science, Royal Institute of Technology
S-100 44, Stockholm, Sweden

Abstract.

In this expository paper the progress in factorization of large integers since the introduction of computers is reported. Thanks to theoretical advances and refinements, as well as to more powerful computers, the practical limit of integers possible to factor has been raised considerably during the past 20 years. The present practical limit is around $10^{75}$ if supercomputers are used and if much computer time is available.

Key Phrases: Integer factorization.

1. Introduction.

The art of decomposing large integers into prime factors has advanced considerably during the last 20 years. It is the advent of high-speed computers that has rekindled interest in this field. This development has followed several lines. In one of these, already existing theoretical methods and known algorithms have been carefully analyzed and perfected. As an example of this work we mention Michael Morrison and John Brillhart's analysis of an old factorization method, the continued fraction algorithm, going back to ideas introduced already by Legendre. This work, supplemented by the quadratic sieve technique, introduced by Carl Pomerance, has led to the construction of the most efficient practically applicable general method so far implemented for factoring large numbers. The practical upper limit for this method with today's computers is the factorization of integers having around 75 digits.

The progress achieved in this area is due also to another line of development, namely the introduction of new ideas. We shall discuss two of these new methods, the method of J. M. Pollard, including a refinement by Richard Brent, and Pollard's $p - 1$-method.

2. General factorization methods.

Even if there exists such a thing as "the most efficient general factorization
algorithm available" at a given time, for practical reasons this algorithm can only be used on "hard cases," i.e. on numbers which cannot be factorized by any simple means. This is because the general factorization methods (which do not take any advantage of a possible special structure of the number to be factorized) and their computer implementations produce a factorization in a time which is virtually independent of the size of the factors or of any special structure of the number. Thus it would be a large waste of computer time to apply such an algorithm only to find a factor that could have been found rapidly by some other means, e.g. a small factor which could have been found by trial division with the small primes. As a matter of fact, efficient factorization of (many) large numbers, with reasonable average running time, calls for a good strategy for factorization, in which not only the most efficient general algorithm, but other methods are involved as well.—In order to describe such a strategy we cannot restrict ourselves describing modern factorization methods only, but we have also to dwell shortly upon some older, well-known methods for factorization as well.

It will also be a pre-requisite for all factorization methods that any number \( N \) subjected to factorization has been proved composite and that any factor found is immediately removed by division, thereby reducing the size of the number remaining to factor.

3. Running times.

The amount of computational work, \( R(N) \), needed to execute some factorization algorithm on the number \( N \), depends on \( N \). Of importance for running time analysis of the algorithm are

\[
\begin{align*}
(1) & \quad \text{the maximal running time} \quad M(N) = \max_{x \leq N} R(x), \\
(2) & \quad \text{the average running time} \quad A(N) = \frac{1}{2\delta + 1} \sum_{N-\delta}^{N+\delta} R(x),
\end{align*}
\]

where the average is taken over a suitably large interval about \( N \). It may be convenient to take the average only over such numbers \( N \) which remain after some preliminary attempts to factor \( N \) have been made.

The dependence of these functions on \( N \) is often expressed as a function of the smallest prime factor of \( N \), or of the second largest prime factor of \( N \), or of some other quantity depending on \( N \).

4. Trial division.

This is the simplest and oldest method for factorization. It is used for finding small factors. Simply divide \( N \) successively by the primes \( p_1 = 2, p_2 = 3, \ldots \)