changing registers ("copying"), memory access, etc. Note that this means that we would use the very same time estimate if we were multiplying a k-bit binary expansion of a fraction by an ℓ-bit binary expansion; the only additional feature is that we must note the location of the point separating integer from fractional part and insert it correctly in the answer.

In the second place, if we want to get a time estimate that is simple and convenient to work with, we should assume at various points that we're in the "worst possible case." For example, if the binary expansion of m has a lot of zeros, then ℓ' will be considerably less than ℓ. That is, we could use the estimate Time(multiply k-bit integer by ℓ-bit integer)$< k \cdot$(number of 1-bits in m). However, it is usually not worth the improvement (i.e., lowering) in our time estimate to take this into account, because it is more useful to have a simple uniform estimate that depends only on the size of m and n and not on the particular bits that happen to occur.

As a special case, we have: Time(multiply k-bit by k-bit)$< k^2$.

Finally, our estimate $k\ell$ can be written in terms of n and m if we remember the above formula for the number of digits, from which it follows that $k = [log_2 n] + 1 \leq \frac{log\, n}{log\, 2} + 1$ and $\ell = [log_2 m] + 1 \leq \frac{log\, m}{log\, 2} + 1$.

Example 6. Find an upper bound for the number of bit operations required to compute $n!$.

Solution. We use the following procedure. First multiply 2 by 3, then the result by 4, then the result of that by 5,..., until you get to n. At the $(j-1)$-th step $(j = 2, 3, \ldots, n-1)$, you are multiplying $j!$ by $j+1$. Hence you have $n-2$ steps, where each step involves multiplying a partial product (i.e., $j!$) by the next integer. The partial products will start to be very large. As a worst case estimate for the number of bits a partial product has, let's take the number of binary digits in the very last product, namely, in $n!$.

To find the number of bits in a product, we use the fact that the number of digits in the product of two numbers is either the sum of the number of digits in each factor or else 1 fewer than that sum (see the above discussion of multiplication). From this it follows that the product of n k-bit integers will have at most nk bits. Thus, if n is a k-bit integer — which implies that every integer less than n has at most k bits — then $n!$ has at most nk bits.

Hence, in each of the $n-2$ multiplications needed to compute $n!$, we are multiplying an integer with at most k bits (namely $j+1$) by an integer with at most nk bits (namely $j!$). This requires at most nk^2 bit operations. We must do this $n-2$ times. So the total number of bit operations is bounded by $(n-2)nk^2 = n(n-2)([log_2 n] + 1)^2$. Roughly speaking, the bound is approximately $n^2(log_2 n)^2$.

Example 7. Find an upper bound for the number of bit operations required to multiply a polynomial $\sum a_i x^i$ of degree $\leq n_1$ and a polynomial $\sum b_j x^j$ of degree $\leq n_2$ whose coefficients are positive integers $\leq m$. Suppose $n_2 \leq n_1$.

Solution. To compute $\sum_{i+j=\nu} a_i b_j$, which is the coefficient of x^ν in the product polynomial (here $0 \leq \nu \leq n_1 + n_2$) requires at most $n_2 + 1$ multi-