$(r/s)^{r/s}e^{ks}$ with respect to s, or equivalently, minimizing its log, which is $\frac{r}{s}\log\frac{r}{s} + ks$. Thus, we set

$$0 = \frac{d}{ds}\left(\frac{r}{s}\log\frac{r}{s} + ks\right) = -\frac{r}{s^2}\left(\log\frac{r}{s} + 1\right) + k \approx -\frac{r}{s^2}\log\frac{r}{s} + k,$$

i.e., we choose s in such a way that ks is approximately equal to $\frac{r}{s}\log\frac{r}{s}$, in other words, in such a way that the two factors in $(r/s)^{r/s}e^{ks}$ are approximately equal. Because k is a constant, it follows from the above approximate equality that s^2 has the same order of magnitude as $r\log(r/s) = r(\log r - \log s)$, which means that s has order of magnitude between $\sqrt{r}$ and $\sqrt{r\log r}$. But this means that $\log s$ is approximately $\frac{1}{2}\log r$, and so, making the substitution $\log s \approx \frac{1}{2}\log r$, we transform the above relation to:

$$0 \approx -\frac{r}{2s^2}\log r + k, \qquad \text{i.e.,} \qquad s \approx \sqrt{\frac{r}{2k}\log r}.$$

With this value of s, we now estimate the time. Since the two factors $(r/s)^{r/s}$ and e^{ks} are approximately equal for our optimally chosen s, the time estimate simplifies to $O(e^{2ks}) = O(e^{\sqrt{2k}\sqrt{r\log r}})$. Replacing the constant $\sqrt{2k}$ by C, we finally obtain the following estimate for the number of bit operations required to factor an r-bit integer n:

$$O\left(e^{C\sqrt{r\log r}}\right).$$

The above argument was very rough. We made no attempt to justify our simplifications or bound the error in our approximate equalities. In addition, both our algorithm and our estimate of its running time are probabilistic.

Until the advent of the number field sieve very recently (see the remark at the end of §5), all analyses of the running time of the best general-purpose factoring algorithms known led to estimates of the form $O\left(e^{C\sqrt{r\log r}}\right)$. In some cases, the estimates were proved rigorously, and in other cases they relied upon plausible but unproved conjectures. The main difference between the time estimates for the various competing algorithms was the constant C in the exponent. In this respect the factoring problem has had a history quite different from the primality problem considered in §1, where improvements in running time (especially of deterministic primality tests) have been dramatic. For a detailed survey and comparison of the factoring algorithms that were known in the early 1980's, see Pomerance's 1982 article cited in the references below.

Remark. Since $r = O(\log n)$, the above time estimate can also be expressed in the form

$$\text{Time(Factor } n) = O\left(e^{C\sqrt{\log n \log\log n}}\right).$$