Algorithmic Analysis

Generally, it's OK to disregard lower-order terms when doing worst-case analysis on algorithmic complexity. For asymptotically large input values, dropping these terms loses very little predictive power. The following notations are ways to describe the asymptotic behavior of functions, or how a function grows when input size n becomes very large. Big-Oh and Big-Theta notation are most commonly used.

$$\begin{aligned}
\Theta(g(n)) &= \{f(n) \mid \exists (c_1, c_2, n_0) \in \mathbb{R}^+ \mid \\
&\qquad \forall n \geq n_0, \quad c_1 g(n) \leq f(n) \leq c_2 g(n)\} && (1)\\
O(g(n)) &= \{f(n) \mid \exists (c, n_0) \in \mathbb{R}^+ \mid \forall n \geq n_0, \quad f(n) \leq c_1 g(n)\} && (2)\\
\Omega(g(n)) &= \{f(n) \mid \exists (c, n_0) \in \mathbb{R}^+ \mid \forall n \geq n_0, \quad cg(n) \leq f(n)\} && (3)\\
o(g(n) &= \left\{ f(n) \mid \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \right\} && (4)\\
\omega(g(n)) &= \left\{ f(n) \mid \lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty \right\} && (5)
\end{aligned}$$