What is practically feasible: a challenging question. Some algorithms are practically feasible, in the sense that they require reasonable time for inputs of reasonable length. Other algorithms are not practically feasible. How can we describe what is practically feasible in precise terms?

The best known approximation to this definition is the notion of feasibility in theoretical computer science, according to which an algorithm is feasible if there exists a polynomial \( P(n) \) such that for each input \( x \) of length \( n \), the computation time \( t(x) \) is bounded by \( P(n) \).

This definition can be described even easier if we use the notion of worst-case complexity \( T(n) \) – defined as the largest computation time \( t(x) \) on all inputs \( x \) of length \( \text{length}(x) = n \): \( T(n) = \max\{t(x) : \text{length}(x) = n\} \). In these terms, feasibility means that for all \( n \), the time \( T(n) \) is smaller than or equal to \( P(n) \).

In many cases, this definition captures the meaning of practical feasibility, but not always. For example, if the computation time \( T(n) \) is \( 10^{100} \times n \) for all \( n \):

- then, as a linear function, \( T(n) \) is a polynomial, thus the corresponding algorithm is feasible according to the formal definition,
- but of course, even for \( n = 1 \), the time \( 10^{100} \times n \) is not practically feasible.

So how can we describe practical feasibility?

Let us use fuzzy approach. The above informal description of practical feasibility uses the notion of what is reasonable. This notion is not precise: some lengths and some times are feasible to some extent. It is not like all the lengths up to some constant \( N \) are reasonable, while an input of length \( N + 1 \) is already not reasonable.

In this sense, the notion “reasonable” is similar to many other commonsense notions like “small”. To describe the meaning of such notions:

- We can assign, to each possible length \( n \), the degree \( L(n) \) (ranging from 0 to 1) to which this input length is reasonable: 1 means that we are absolutely sure than \( n \) is a reasonable length, 0 means that we are absolutely sure that \( n \) is not a reasonable length, and values between 0 and 1 correspond to intermediate degrees of confidence.
- Similarly, we can assign, to each possible time \( t \), the degree \( R(n) \) to which this time is reasonable.

This approach – going back to Lotfi Zadeh – is known as fuzzy approach.
An algorithm with worst-case complexity $T(n)$ is feasible if for every $n$ for which the corresponding length is reasonable, the time $T(n)$ is also reasonable, i.e., for all $n$, we have the implication $L(n) \rightarrow R(T(n))$. In other words, we have the implication corresponding to $n = 1$ and the implication corresponding to $n = 2$, etc.:

$$(L(1) \rightarrow R(T(1))) \& (L(2) \rightarrow R(T(2))) \& \ldots \& (L(n) \rightarrow R(T(n))) \& \ldots$$

Fuzzy approach allows us to describe the degree to which this statement is true:

- the easiest way to describe the degree to which the statement $A \& B$ is true if as $\min(A, B)$, and
- the easiest way to describe the degree to which the implication $A \rightarrow B$ is true is $\max(B, 1 - A)$.

The last expression comes from the fact that:

- in the usual logic, $A \rightarrow B$ is equivalent to $B \lor \neg A$,
- the degree of belief in “not A” is naturally described as $1 - A$, and
- the degree of belief in “or” of two statements can be estimated as the maximum of the degrees of belief in the two statements.

By using these formulas, we can describe:

- the degree to which statement $L(n) \rightarrow R(T(n))$ is true by the expression
  $$\max(R(T(n)), 1 - L(n))$$
  and
- the resulting degree $d$ to which an algorithm with time complexity $T(n)$ is feasible as
  $$d = \min\{\max(R(T(n)), 1 - L(n)): n = 1, 2, \ldots\}$$

**How can we compute this degree.** The larger $n$, the smaller the degree $L(n)$ that $n$ is a reasonable length, so the larger the difference $1 - L(n)$.

On the other hand, the larger $n$, the larger $T(n)$ and thus, the smaller the degree $R(T(n))$ that the time $T(n)$ is reasonable.

Thus, if $R(T(n_0)) < 1 - L(n_0)$ for some $n_0$, then this inequality is true for all larger values $n$ as well: since with an increase in $n$, the value $R(T(n))$ will become even smaller and the value $1 - L(n)$ will become even larger. For all these values, the maximum is thus equal to $\max(R(T(n)), 1 - L(n)) = 1 - L(n)$. This value increases with $n$, so its minimum is attained for $n = n_0$.

Similarly, if $R(T(n_1)) > 1 - L(n_1)$ for some $n_1$, then this inequality is true for all smaller values $n$ as well: since with a decrease in $n$, the value $R(T(n))$ will become even larger and the value $1 - L(n)$ will become even smaller. For all these values, the maximum is thus equal to $\max(R(T(n)), 1 - L(n)) = R(T(n))$. This value decreases with $n$, so its minimum is attained for $n = n_1$.

Thus, to find the degree $d$, it is sufficient to consider the first value $f$ for which $R(T(f)) < 1 - L(f)$, and to consider only two value $n = f$ and $n = f - 1$:

$$d = \min(R(T(f - 1)), 1 - L(f)),$$