T.C.A.S.: **T**CAS **C**an **A**lways **S**olve

Numerical Analysis: a Concrete Computational Approach to Limit Evaluation

Roberto Alessandro Bertolini

Liceo Nervi Ferrari - Morbegno

Abstract

Evaluating limits by hand can become a trivial task with a bit of exercise, but a normal computer is generally incapable of proceeding intuitively and needs a reliable algorithm in order to be able to reach consistently the same result. While some purely heuristic or naive approaches might, at first glance, seem good enough, they tend to quickly fall apart in real world conditions. TCAS is a portable universal C implementation of the Gruntz Algorithm [1], which at the present day is the most efficient and reliable way of evaluating limits in the exp-log field of operations.

Contents

1 Introduction

1.1 The Limit of a Function

The limit of the function $f : \mathbf{R} \to \mathbf{R}$ is defined as the following:

Definition 1.

$$\lim_{x \to x_0} f(x) = l$$

If and only if $\forall \varepsilon > 0 \ \exists \ \delta(\varepsilon) \mid \forall x \in D_f, 0 < \mid x - x_0 \mid < \delta \implies \mid f(x) - l \mid < \varepsilon$, where $x_0, \varepsilon \in \mathbf{R}$

1.2 Polish Notation

Jan Lukasiewicz devised the so-called Polish notation [3] in 1924; it is a prefix notation, where the operator precedes its operands. As long as the number of operands is predefined, there can't be an ambiguity in the order of evaluation, so this notation doesn't strictly require parenthesis. Consider the following expression written without parenthesis:

$$8 \times 4 + 3 \times 2 - 6$$

Depending on where the parenthesis are placed, it can evaluate to different results:

$$(8 \times 4 + 3) \times (2 - 6) = -140; \quad (8 \times (4 + 3) \times 2) - 6 = 106$$

Now consider a similar expression written using polish notation:

$$\times \times 8 + 4 \ 3 - 2 \ 6 = -224$$

It necessary evaluates to a single result. This makes parsing the expression into an abstract syntax tree much easier, as the parser doesn't have to make educated choices about its interpretation.

1.3 The Shortcomings of the Naive Approach

Evaluating a limit might seem easy for a computer, as it should be able to continuously approximate the result with a smaller ε until the rounding error is acceptable enough, but can this always work? Consider the following function:

$$f : y = \frac{1}{x^{\ln \ln \ln \ln \frac{1}{x} - 1}} \tag{1.3}$$

Its graph is the following: fig. 1, where it seems that for values of x closer to zero the function tends to zero, yet if we compute the limit: $\lim x \to 0^+ f(x) = +\infty$, so the function should have a vertical asymptote in zero. We just can't see it in the graph because it becomes visible for $x \approx 4.29 \times 10^{-1656521}$. A computer trying to approximate the result would have stopped long before this value, returning zero.

What if we try instead to recursively apply L'Hôpital's rule [2] until we reach a clear result? Consider the following functions:

$$f(x) = e^x + e^{-x}$$

$$g(x) = e^x - e^{-x}$$

$$\lim_{x\to\infty}\frac{f(x)}{g(x)}=\lim_{x\to\infty}\frac{f''(x)}{g''(x)}=\lim_{x\to\infty}\frac{e^x+e^{-x}}{e^x-e^{-x}}$$

So recursively applying L'Hôpital's rule would not work in this case.

2 The Gruntz Algorithm

2.1 Manipulating the Most Rapidly Varying Subexpressions

Definition 2. $g(x)$ is said to be a subexpression of $f(x)$ if the evaluation of $f(x)$ causes the evaluation of $g(x)$. This is represented with the following notation:

$$g(x)\lhd f(x),\text{ if } g(x)\text{ is a subexpression of } f(x)$$
$$g(x)\not\lhd f(x),\text{ if } g(x)\text{ is not a subexpression of } f(x)$$

Definition 3. Two subexpressions $g(x)$ and $h(x)$ can be compared based on their comparability class, which describes how rapidly they vary.

$$f(x)\prec g(x)\text{ if and only if }\lim_{x\to\infty}\frac{\ln|f(x)|}{\ln|g(x)|}=0$$
$$f(x)\succ g(x)\text{ if and only if }\lim_{x\to\infty}\frac{\ln|f(x)|}{\ln|g(x)|}\neq 0\in\mathbf{R}$$

Definition 4. The MRV set is the set that contains all the subexpressions of the highest comparability class.

$$mrv(f(x))=\begin{cases}\{\} & \text{if } x\not\lhd f(x)\\ \{g(x)\mid g(x)\lhd f(x)\wedge(\nexists\ h(x)\lhd f(x)\mid h(x)\succ g(x)))\}\end{cases}$$

The first step of the Gruntz algorithm is recursively computing the MRV set of the limit that needs evaluating. If empty, the limit is indipendent of x. Once found, all the subexpressions contained in it must be rewritten as a function of w, which itself has to respect the following expression:

$$\lim_{x\to\infty}w(x)=0+$$

The most common w is $w(x)=e^{-x}$.

2.2 Power Series Representation

Definition 5. The Maclaurin series of any function is defined as the following:

$$\sum_{n=0}^{\infty}\frac{f^{(n)}(0)}{n!}x^n$$

For the algorithm we need the first non-zero term of the Maclaurin series of the function after the w substitution, which will be in the form $A(x)w^b$.

The final result of the limit will be the following:

$$\lim_{x \to x_0} f(x) = \lim_{x \to 0} w(x) = \begin{cases} 0 & \text{if } b > 0 \\ +\infty & \text{if } b < 0 \\ \lim_{x \to 0} A(x) & \text{if } b = 0 \end{cases}$$

2.3 Caveats and Limitations

The Gruntz algorithm, while mathematically proved to be always valid in the exp-log field of operations, has some caveats that, if not handled correctly, might prevent it from reaching the correct result, or reaching a result at all. Firstly, before computing the MRV, the limit has to be rewritten so that it tends to $+\infty$, applying some smart substitutions. Computing the correct MRV set is a complex operation, which involves recursively evaluating limits, which are proven to be of lower complexities only if simplifications are applied all throughout the process. If the MRV is of a lower comparability class compared to x, the limit has to be rewritten so that the MRV increases its comparability class, usually applying the following definition:

Definition 6.

$$\lim_{x \to +\infty} f(x) = +\infty \bigwedge \lim_{x \to +\infty} g(x) = +\infty \implies \lim_{x \to +\infty} f(g(x)) = +\infty$$

3 T.C.A.S.

3.1 Expression Parsing

3.2 Finding the MRV

3.3 Evaluating the Result

3.4 Performance Considerations

References

[1] Dominik Wolfgang Gruntz. *On Computing Limits in a Symbolic Manipulation System*. 1996. URL: https://doi.org/10.3929/ethz-a-001631582.

[2] Wikipedia contributors. *L'Hôpital's rule — Wikipedia, The Free Encyclopedia*. [Online; accessed on 25-May-2021]. 2021. URL: https://en.wikipedia.org/wiki/L%5C%27H%5C%C3%5C%B4pital%5C%27s_rule.

[3] Wikipedia contributors. *Polish notation — Wikipedia, The Free Encyclopedia*. [Online; accessed on 25-May-2021]. 2021. URL: https://en.wikipedia.org/wiki/Polish_notation.

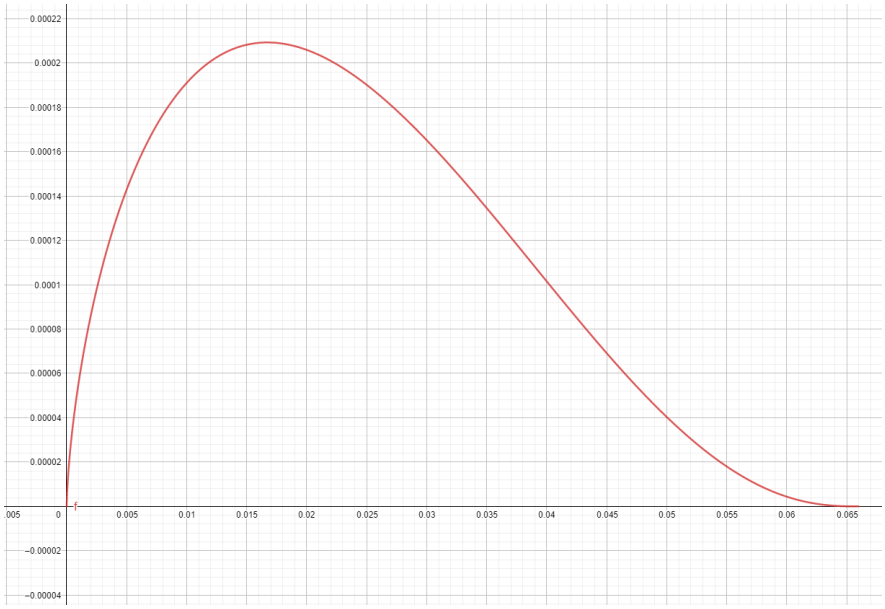

Figure 1: The graph of the function (1.3)