A Code Search Engine for Go

Harrison Brewton
Advised by Aws Albarghouthi
April, 2020

"All programming language proofs are by induction"

1 Background

2 Overview

3 Type2Metric

3.1 Levenshtein Combinator

The Levenshtein edit distance is a commonly used metric for strings. It gives the number of substitutions, deletions, or additions to edit one string into another string [1]. It is well known that the normal Levenshtein distance on strings satifies the properties of a metric. There are a couple of assumptions that the Levenshtein metric makes: that all substitutions are of the same cost, that deletions and additions are of the same cost and the same cost as a substitution. While this is a safe assumption for strings of characters, it is not genereral, as there might be better substitutions than others. For example, it is clear that in a real sense $[1, 2, 3]$ is closer to $[2, 2, 3]$ than it is to $[10, 2, 3]$, as a 1 is closer to a 2 than a 10; however, the Levenshtein edit distance would be the same from the first string to the second string (1). To that end we define the Levenshtein Combinator.

Definition 1 (Levenshtein Combinator)**.** *Assume that $d : (T \times T) \to \mathbb{R}^{\geq 0}$ is a metric. Suppose further that κ is any positive real. Then we say the Levenshtein Combinator of d is $Lev_d : (List\quad T \times List\quad T) \to \mathbb{R}^{\geq 0}$. Such that it is defined on lists of T. We define the combinator as:*

$$Lev_d(a, b) = \begin{cases} \kappa|a[1:]| & |b| = 0 \\ \kappa|b[1:]| & |a| = 0 \\ min \begin{cases} Lev_d(a[1:], b) + \kappa \\ Lev_d(a, b[1:]) + \kappa \\ Lev_d(a[1:], b[1:]) + d(a[0], b[0]) \end{cases} & \textit{otherwise.} \end{cases}$$

The Levenshtein Combinator runs almost the same as the traditional Levenshtein edit distance. The only differences are the generalizations of the κ from 1, and the addition of the metric between elements of the list. In fact, the original Levenshtein distance can be recovred by substituting 1 for κ, and the characteristic function $1_{a \neq b}$ as the d metric. We will now show that the Levenshtein combinator of a metric is itself a metric.

Identity of indiscernibles and symmetry are pretty obvious, and so are omitted. The triangle inequality also follows in a similar manner as the proof of traditional levenshtein [2]. Suppose we are considering a transformation between lists X and Z, that is a series of insertion, deletions, and substitutions. A transformation from X to some Y followed by a transformation from X to some Z is a transformation from X to Z. As the transformation from X to Z is the minimal transformation, it follows $d(X, Z) \leq d(X, Y) + d(Y, Z)$.

3.2 Linear Combinator

We will now briefly show that the linear combination of metrics is a metric. Suppose we have a metric $d(a, b) = \sum c_i d_i(a, b)$, where c_i is a positive real number, and d_i is a metric. Symmetry follows clearly.

Identity of indescernables follows as each term goes to zero, leaving the sum at zero. The triangle ineqaulity follows $d(a,b) = \sum c_i d_i(a,b) \leq \sum c_i d_i(a,c) + c_i d_i(a,b) = d(a,c) + d(b,c)$.

3.3 From Types To Metrics

$$\frac{x : \text{List} \quad T \qquad y : \text{List} \quad T}{\text{Lev}_T(x,y)}\ \textsc{Lists} \qquad \frac{x : (T_1, T_2, \ldots, T_n) \qquad y : (T_1, T_2, \ldots, T_n)}{\displaystyle\sum_{i=0}^{n} c_i d_{T_i}(x_i, y_i)}\ \textsc{Products}$$

$$\frac{x : T_1|T_2|\ldots|T_n \qquad y : T_1|T_2|\ldots|T_n}{1_{\text{type}(x)=\text{type}(y)} d_{\text{type}(x)}(x,y) + 1_{\text{type}(x)\neq\text{type}(y)}\infty}\ \textsc{Sums}$$

4 Example: Go Code Look Up

4.1 Generics

5 Metric Trees

6 Implementation

7 Related Work

8 Future Work

8.1 Best Approximation

9 Conclusion

References

[1] Wikipedia contributors. levenshtein distance, 2020. [Online; accessed 19-April-2020].

[2] Jeremy Kun. Metrics on words, May 2014.