Revisiting Linguistic Approximation for Computing with Words

N. Marhamati, P. Patel, Y. Althobaiti, E. S. Khorasani, and **S. Rahimi**
Department of Computer Science
Southern Illinois University Carbondale

Abstract

Any computing with words (CW) system is required to assign a phrase in natural language to the fuzzy values it provides as its output. This paper explores different linguistic approximation methods for CW systems. The outputs of these methods are evaluated through various measures such as fuzziness, specificity, validity, and sigma-count. We illustrate that certain linguistic methods may result in complex and incomprehensible phrases in natural language. Some might even include an invalid linguistic term in their linguistic approximation.

1 Introduction

Zadeh introduced the concept of computing with words (CW) over a decade ago(Zadeh 1996),(Zadeh 2000) and highlighted the significant role that perceptions are capable of in development of science and technology. CW interprets propositions in natural language as a relationship between a linguistic variable (V) and a linguistic term (T). This relationship is called generalized constraint (GC) and is represented as $VisrT$ where *r* determines the modality of the relationship.

So far, there have been minimal efforts to implement a CW engine in the soft computing community (Mendel 2001),(Khorasani et al. 2011). To our knowledge, CWShell (AKA CWJess), built on top of Jess rule based engine (Khorasani et al. 2011), is the only computing with words expert system shell implementation. CWShell executes a sequence of inference steps for producing fuzzy results as an output of the user's query. CWShell should be capable of translating the results of its reasoning from a fuzzy set representation into an appropriate and understandable word or a natural language sentence. In other words, it should be able to decode a fuzzy value to a linguistic term that can be evaluated by a human decision maker. Fig. 1 illustrates an overall view of a general CW system. In this paper, we explore different methods for retranslating the fuzzy values applicable to the paradigm of CW. The retranslation refers to the decoding of a GC to a linguistic proposition in natural language, as shown in Fig. 1. This process of associating propositions in natural language, based on the semantic meaning of the fuzzy values, has been referred to as linguistic approximation (LA).

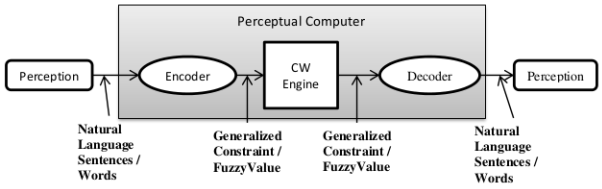

Figure 1: Retranslation of fuzzy values to words in a CW system

The study in this paper is an attempt towards discovering an appropriate LA method for CW. We evaluate applicability of several methods to the CWShell system through a case study. Our case study is the real world problem of calculating a person's insurance cost (Khorasani et al. 2012). In section 2, we categorize existing LA methods and in Section 3, we apply these methods to the case study and assess performance of each LA method based on several criteria, such as fuzziness, specificity, validity, and sigma-count.

2 Linguistic Approximation Methods

Formally, given a collection of terms in natural language associated with fuzzy subsets in the universe X, a linguistic approximation algorithm maps a fuzzy set to the collection of terms (Eshragh and Mamdani 1979). In terms of computing with words, it is a retranslation process of replacing the proposition $VisZ$ with $VisT$, where T is a collection of linguistic terms predefined for X. We categorize LA algorithms into three groups as follows in this section.

2.1 Simple LA

A simple LA method compares the fuzzy value directly with the basic terms of a linguistic variable. Let Z be the fuzzy set that needs to be approximated using the term set T, called atomic linguistic terms. The terms in T have pre-existing natural language labels assigned to it. A simple LA algorithm compares Z with all the terms in T and selects the one with the highest similarity measure.

To perform the comparison, some indices are required to determine similarities between the fuzzy value and the basic terms in the database. These indices may either measure

the distance, compatibility, or the similarity of sets , and as characteristics of fuzzy sets, they may consider the shape or proximity of the sets. A similarity relation, $R(A, B)$, is defined on the interval $[0, 1]$ and determines the similarity of fuzzy sets A and B.

There has been a vast amount of effort by researchers to define and evaluate the performance of such indices. In the literature, approximately 50 expressions have been introduced to compare fuzzy sets. We have selected some of these methods, which either are intensively cited or have recently been proposed, and applied them to retranslation of fuzzy sets to words in CWShell engine. For an exact definition of these comparison methods, please refer to (Zwick, Carlstein, and Budescu 1987).

Tversky's similarity function This approach determines the common and distinct features of objects to compare objects with each other. One of the very popular methods that can be considered a particular case of Tversky's ratio model is Jaccard's index which can be calculated by the scalar cardinality of the intersection and union of the two fuzzy sets.

Bonissone's distance This method approaches the problem from a pattern recognition point of view and defines some features for each fuzzy set which are assigned to each set in a vector. This method consists of two stages where the first stage employs weighted Euclidean distance (d_1) while the Bhattacharyya distance is used in the second stage (d_2).

Wenstøp's method Wenstøp represents each fuzzy set with two parameters, centroid and cardinality. Similar to the first step of Bonissone's approach, this approach uses the regular Euclidean distance between the two corresponding vectors to determine similarity of sets.

VSM measure Vector Similarity Measure (VSM) (Wu and Mendel 2008) employs a vector composed of two different measures to evaluate the similarity of two fuzzy sets. To compare shapes of sets, VSM shifts the sets such that their COGs (Center of Gravity) coincide, and computes Jaccard's measure for the shifted sets.

A very basic LA algorithm would compare all the atomic terms, say T, with a fuzzy value output of a CW system, say Z, using one of the similarity measures, $R(A, B)$, described above. The term with the best match (highest similarity value) is selected as LA for the fuzzy value Z . Although this approach is computationally efficient, it might not provide a good approximation for Z, particularly when the similarity between Z and the selected term is low.

2.2 LA using Modifiers

Fuzzy modifiers, or hedges, can be used for better approximation. The most commonly discussed modifiers in the literature include NOT, VERY, MORE OR LESS, INDEED, etc. To apply the modifiers to basic atomic terms, we can generate the expression of the form $< linguistic\, modifier >< atomic\ term >$. Next, the closest LA expression can be selected by comparing all the pairs of atomic terms and linguistic modifiers with the fuzzy value, Z, and selecting the one with the largest value of similarity measure. We call this approach $ModifiedLA$ algorithm from here on.

This simple approach can be optimized by using the method called piecewise decomposition, in which the fuzzy value is decomposed into several segments. Each segment is then combined with the modifiers and used for comparison with the fuzzy subsets in S to generate linguistic expressions. These individual linguistic expressions are combined using the connectives C_i, such as AND and OR, based on the properties of their segments. The resultant linguistic expression would be of the form $< linguistic\, modifier >< atomic\ term\ > [C_i\ <\ linguistic\, modifier\ >< atomic\, term\ >]^*$, where the expression in square brackets can have zero or more occurrences.

Algorithm 1 $PiecewiseLA(Z)$

```
segmentsZ = decomposition of fuzzy value Z into segments
for all segZ in segmentsZ do
    modifiedSentence = ModifiedLA(segZ)
    if bestSentence is empty then
        bestSentence = modifiedSentence
    else
        determine C_i
        bestSentence = bestSentence C_i modifiedSentence
    end if
end for
return bestSentence
```

Algorithm 1 summarizes the approach for piecewise decomposition for generating modified LA sentences. The fuzzy value Z is decomposed into segments $segmentsZ$. There are multiple ways to decompose fuzzy sets into segments (refer to (Dvořák 1999),(Eshragh and Mamdani 1979)).

2.3 LA using Quantifiers

In most real situations, even the best linguistic approximation method can provide only a partial match. Hence, some information is inevitably lost in every approximation method. To deal with this problem, the quantified linguistic approximation method (Kowalczyk 1999) suggests the addition of another attribute, i.e. a linguistic quantifier, to an LA method in order to determine what proportion of the fuzzy value under approximation matches with the outcome of the LA method. In other words, the quantified LA method allows us to map a fuzzy value Z into $QZisT$ where Q is a linguistic quantifier and T is a linguistic term obtained by using an LA method.

The fuzzy quantifier Q signifies the portion of Z belonging to T. Q is a mapping from $[0, 1]$ to $[0, 1]$ and can be expressed by using relative sigma count or cardinality. Relative cardinality determines the portion of elements from Z belonging to A and can be formulated as

$$c = \sum Count(A/Z) = \sum_{x \in X} \mu_{A \cap Z}(x) / \sum \mu_A(x) \qquad (1)$$

where $\mu_A(x)$ stands for membership value of x in A. The value of the relative sigma count, c, is henceforth used to determine the appropriate label for the linguistic quantifier Q from a set of predefined fuzzy quantifiers. The fuzzy quantifier which has the highest degree of membership at c is chosen to represent Q.

Algorithm 2 $QuantifiedLA(Z)$

$bestSentence = PiecewiseLA(Z)$
$A =$ fuzzy set represented by $bestSentence$
$c = \Sigma Count(A/Z)$
$m = 0$
for all q in $QSet$ **do**
  if $\mu_q(c) \geq m$ **then**
    $m = \mu_q(c)$
    $Q =$ label of q
  end if
end for
$bestSentence = QZ$ is $bestSentence$
return $bestSentence$

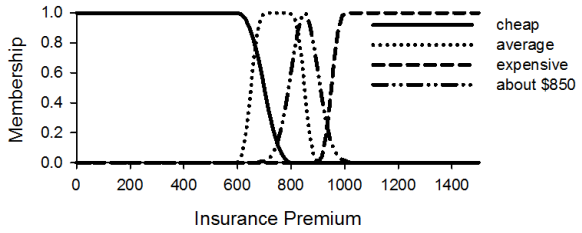

Figure 2: Predefined linguistic terms for *insurance premium*

Algorithm 2 summarizes the quantified linguistic approximation method. This algorithm takes the outcome of the piecewise LA method (Algorithm 1), calculates the relative sigma-count, and finds a linguistic quantifier which can best describe how much of the fuzzy value under approximation is covered by the outcome of the piecewise LA method. The $Qset$ denotes the set of predefined fuzzy quantifiers.

3 Towards LA for CWShell

In order to evaluate the LA techniques, we applied these methods to the output fuzzy value of an insurance premium problem using CWShell.

3.1 Case Study: Insurance Premium

The case-study is about finding the auto-mobile insurance premium, provided some information about a car and its owner such as age, residency, driving records, price of the vehicle and etc. Details of the case study can be found in (Khorasani et al. 2012)):

The linguistic term sets for the linguistic variable *insurance premium*, as provided by the company's experts, is plotted in Fig. 2. The output (the individual's insurance premium) as computed by CWShell inference engine for the case study is a fuzzy set shown in Fig. 3. Although its shape is similar to a simple trapezoid, linguistic approximation for

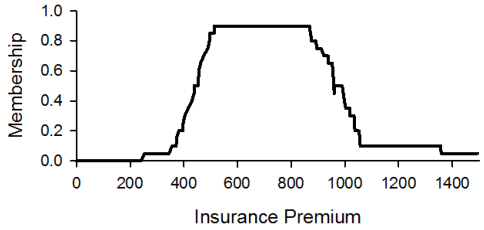

Figure 3: Output of CWShell as a fuzzy set

Table 1: Similarity of the fuzzy set *insurance premium* with predefined linguistic terms according to different measures

Linguistic Term	VSM	Jaccard	Wenstøp/d_1	Bonissone-d_2
cheap	0	0.2776	1.7378e4	0.6521
average	1.1e-81	0.3540	3.2626e4	0.5761
expensive	0	0.0549	2.4310e3	0.8934
about $850	1.4e-255	0.2327	4.0093e4	0.6082

this set is a challenging problem since it overlaps with three different terms.

The result of applying the simple LA method category (section 2.1) to this case study, using different indices, is represented in Table 1. Results of Wenstøp and Bonissone-d_1 are exactly the same so they are shown under the same column. For VSM and Jaccard similarity measure, the output of LA would be the term with the largest value. In contrast, Wenstøp and Bonissone (d_1, and d_2) select the term with the least index, since they measure distance between fuzzy sets. The VSM, Jaccard, and Bonissone-d_2 assign the term *average* to the output plotted in Fig. 3 while the Wenstøp and Bonissone-d_1 assign the term *expensive*. Result of piecewise LA (section 2.2) for the case study is *"VERY cheap AND EXTREMELY average AND MOREORLESS about $850"*. Applying quantifiers (section 2.3) to this combination would result in the expression *"ALMOST ALL Z are (VERY cheap AND EXTREMELY average AND MOREORLESS)"*

3.2 Evaluation Criteria

We applied specificity and fuzziness of the sets, and validity of assigning a linguistic term to fuzzy values, as the three criteria to evaluate LA results. Detailed explanation on these criteria can be found in (Yager 2004). The other criterion we use is the relative sigma count of the terms with respect to the fuzzy value *insurance premium*.

Following (Yager 2004) work, fuzziness of a fuzzy set A is determined as

$$fuzz(A) = 1 - \frac{1}{n} \sum_{x \in X} |2\mu_A(x) - 1| \quad (2)$$

where n is the number of members of A in universe of discourse X. Fuzziness is a concept that distinguishes members from non-members of a set.

Specificity of a fuzzy set is related to how much information is conveyed by that fuzzy set. It is defined as

$$spf(A) = Max_{x \in X}(\mu_A(x)) - \frac{1}{n-1} \sum_{x \in X, x \neq x^{max}} \mu_A(x) \quad (3)$$

The degree of validity explains the validity of the entailment ($VisrZ \Rightarrow VisrT$), where Z refers to the fuzzy value *insurance premium* and T is substituted by any of the linguistic terms. Degree of validity is calculated according to

$$Deg_{validity}(Z,T) = Min_{x \in X}[I(\mu_Z(x), \mu_T(x))] \quad (4)$$

where there are different methods to define operator I and we use three of those method as follows

$$Lukasiewicz : I(z,t) = Min(1, 1+t-z) \quad (5)$$

$$Goguen : I(z,t) = Min(1, \frac{t}{z}) \quad (6)$$

$$Godel : I(z,t) = \begin{cases} 1 & z \leq t \\ 0 & z > t \end{cases} \quad (7)$$

Table 2: characteristics of fuzzy sets- the fuzzy set insurance premium is shown in Fig. 3 and the rest of sets are plotted in Fig. 2

Linguistic Term	Fuzziness	Specificity	Relative Sigma-Count
insurance premium	0.1943	0.7435	1
cheap	0.0444	0.8889	0.4262
average	0.0444	0.9286	0.9351
expensive	0.0222	0.9500	0.0840
about $850	0.0698	0.9165	0.9689

3.3 Discussion and Conclusion

These criteria can not directly be applied to the result of piecewise LA algorithm since its output is a combination of words and a fuzzy value cannot directly be constructed. If we use AND to aggregate different segments, then the intersection of *very cheap*, *extremely average*, and *more or less about $850* does not cover a significant area of the fuzzy set in Fig.3 and hence is not a good approximation for it.

If the value of fuzziness and specificity of the assigned term is closer to the value of fuzziness and specificity of the original fuzzy set, then it signifies a more accurate LA. Also, the value of relative sigma count approaching to 1 signifies a better LA result. Specificity, fuzziness, and relative sigma count of all the available terms is calculated and compared to the fuzzy value *insurance premium* in Table 2. Fuzziness of all terms is much less than the fuzziness of the *insurance premium's* fuzzy value and its specificity is not that close to any of the terms' fuzzy value. Therefore, none of the single terms is a perfect match for it. Relative sigma count of terms *average* and *about $850* are closer to one.

Table 3 shows validity of assigning different linguistic terms to the target fuzzy value (insurance premium). Only the linguistic terms *average* and *about $850* have positive degree of validity so only these terms are valid to be assigned to the fuzzy value *insurance premium*. Therefore, the result of LA methods based on similarity measures VSM, Jaccard, and Bonissone-d_2 are reasonable. In contrast, the terms *very cheap* and *expensive* in result of piecewise LA algorithm are not proper.

Similarity based LA methods, assign a single term to the target fuzzy value, although combination of the terms *average* and *about $850* seems more reasonable as an LA output. The output of piecewise LA algorithm is a combination of the terms *cheap*, *average* and *about $850*, but term *cheap* is not a valid term. Consequently, *VERY cheap* in the expression of piecewise LA algorithm, not only makes validity of the result questionable, but also adds to the complexity of the output. Moreover, this LA method formulates a strange phrase in the natural language. Using quantifiers, a more accurate result can be generated but again not a decent natural language phrase. Hence these two methods may not be appropriate to be applied in a CW system. Among the successful similarity measures in our case study, Bonissone and VSM are more compels to be implemented, possibly with some modifications and adjustments. The VSM method uses Jaccard's index of shifted fuzzy sets for its first vector element and a proximity measure as its second element. Since VSM is originally designed for interval type-2 fuzzy sets, it compares the shapes of fuzzy sets as well as their proximity. Both shape and proximity affect intersection and union of sets, Hence the Jaccard's index is also influenced by both shape and proximity. But, VSM doesn't benefit from this property of Jaccard's index since VSM applies it to shifted sets and uses it only for shape comparison.

Table 3: Validity of assigning different linguistic terms to the target fuzzy set insurance premium

Linguistic Term	Lukasiewicz	Goguen	Godel
cheap	0	0	0
average	0.92	0.92	0
expensive	0	0	0
about $850	0.92	0.92	0

So the results lead us to the conclusion that the piecewise LA and quantified LA algorithms might add invalid terms to their output and generally result in complex phrases that are not preferable in natural language. On the other hand, simple LA algorithms result in valid matches though not a perfect one.

References

Dvořák, A. 1999. On linguistic approximation in the frame of fuzzy logic deduction. *Soft Computing* 3(2):111–115.

Eshragh, F., and Mamdani, E. 1979. A general approach to linguistic approximation. *International Journal of Man-Machine Studies* 11(4):501–519.

Khorasani, E. S.; Rahimi, S.; Patel, P.; and Houle, D. 2011. Cwjess: Implementation of an expert system shell for computing with words. In *FedCSIS* (2011), 33–39.

Khorasani, E.; Patel, P.; Rahimi, S.; and Houle, D. 2012. An inference engine toolkit for computing with words. *Journal of Ambient Intelligence and Humanized Computing* 1–20.

Kowalczyk, R. 1999. On quantified linguistic approximation. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, UAI'99, 351–358. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Mendel, J. M. 2001. The perceptual computer: An architecture for computing with words. In *FUZZ-IEEE*, 35–38.

Wu, D., and Mendel, J. 2008. A vector similarity measure for linguistic approximation: Interval type-2 and type-1 fuzzy sets. *Information Sciences* 178(2):381–402.

Yager, R. R. 2004. On the retranslation process in zadeh's paradigm of computing with words. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 34(2):1184–1195.

Zadeh, L. 1996. Fuzzy logic= computing with words. *Fuzzy Systems, IEEE Transactions on* 4(2):103–111.

Zadeh, L. 2000. From computing with numbers to computing with wordsfrom manipulation of measurements to manipulation of perceptions. *Intelligent Systems and Soft Computing* 3–40.

Zwick, R.; Carlstein, E.; and Budescu, D. 1987. Measures of similarity among fuzzy concepts: A comparative analysis. *International Journal of Approximate Reasoning* 1(2):221–242.