Journal of Advance Computational Research
Vol. 3 No. 2 (July-December, 2018)

Constraint Propagation Tree for the Realization of a Computing with Word Based Question Answering System

Elham S. Khorasani, Shahram Rahimi
Southern Illinois University Carbondale, Carbondale, IL, 62901

RECEIVED: August 08,2017. Revised June 30, 2018

Abstract: This paper develops a deduction methodology for a domain specific Computing with Word based question answering system. This methodology takes, as input, a knowledge base and a query in form of generalized constraints and organizes the knowledge related to the query in a tree structure, referred to as the constraint propagation tree (CPT). CPT Generates a plan to find the most relevant answer to the query. It also identifies the missing knowledge in the knowledge base and allows improving the answer through establishing an information-seeking dialog with the user. To facilitate the implementation of a CPT, the knowledge related to a query is classified in to three canonical forms and a set of rewriting rules are proposed to convert the data related to a query into one of these forms.

Index Terms: Computing with Words, Question Answering, Backward Reasoning, Constrain Propagation Tree.

I. INTRODUCTION

The Question Answering (QA) systems are regarded as the next generation of the current search engines. They receive a query expressed in natural language, process their knowledgebase, which is also in natural language, and return the most relevant answer to the query. Therefore, QA systems need more complex natural language processing than other types of information retrieval systems. Current question answering systems typically use pattern matching and semantic search methods to extract the candidate answers to a query from a pool of related documents. When the answer vocabulary does not exactly match the query vocabulary, a lexical database such as Wordnet [3] is used to bridge the vocabulary mismatch. Some advance question answering systems, such as Power-Answer [9], also include a reasoning component to return the answers that are not explicitly stated in the corpus but can be inferred from it. Such reasoning component typically utilizes predicate logic to derive the answers which are implicit in the corpus. However, predicate logic is incapable of reasoning with imprecise words inherent in natural language and hence it is very limited in formulating human reasoning.

For example, suppose that the query is to speculate the average price of petroleum for the next year, while the corpus contains information such as: "the average price of petroleum this year was higher than $73 per barrel and it will likely to rise sharply over the course of next year". The predicate logic comes to no conclusion in such case as it fails to perform computations on imprecise words such as: " higher than $73", "likely", and "sharply". Hence, developing a more intelligent question answering system requires an advanced mathematical tool which can model the meaning of imprecise words drawn from natural language and perform reasoning among perceptions. The theory of Computing with Words (CW)[19] , which is rooted in fuzzy set and fuzzy logic, provides such tool.

The core of CW is to view a proposition in natural language as imposing a soft/hard constraint on some attributes and represents it in form of a generalized constraint (GC). In general a GC is in form of:

$$GC : X \text{ is } R$$

Where X is a linguistic (or constrained) variable whose values are constrained by the linguistic (or fuzzy) value R. A linguistic variable can take various forms; it can be a relation (such as: (X, Y)), a crisp function of another variable (such as: $f(Y)$), or it can be another GC. The small r shows the semantic modality of the constraint, that is: how X is related to R. various modalities are characterized by Zadeh, among them are:

- possibility (r = blank): where R denotes the possibility distribution of X, e.g., "X is large.".

Correspondence email address: **elhams,rahimi@siu.edu**

- verity (r = "v"):where R denotes the truth distribution of X, e.g., "(X is large) isv very true".
- identity (r = "="): where X and R are identical variables.
- fuzzy graph (r = "isfg"): where R is a fuzzy estimation of a function. This modality corresponds to a collection of fuzzy if then rules that share the same variables in their premises and consequences.
- probability (r = "p"): where R is the fuzzy probability distribution of X, e.g., "(X is large) isp likely".
- usuality (r = "u"): where R is the typical value of X, e.g., "X isu Large".

A collection of GCs together with a set of logical connectives (such as: and, or, implication, and negation) and a set of inference rules form the generalized constraint language (GCL). The inference rules regulates the propagation of GCs. Table I lists instances of GCL inference rules formulated by Zadeh. Each inference rule has a syntactic part and a semantic part. The syntactic part shows the general abstract form (also called the protoform) of the GCs of the premises and the conclusion of the rule, while the semantic part is a semantic condition which makes the rule valid. The inference rules of CW are adopted and formalized from various fuzzy fields such as: fuzzy probability, fuzzy logic, fuzzy relations, fuzzy quantifiers, and fuzzy arithmetic.

Table I
Instances pf CW Inference Rules [20]. U and V are The Universes of Discourse of X and Y

Inference Rule	*Syntactic part*	*semantic part*
Extension Principle	X_1 is A_1 X_2 is A_2 $\vdots$ X_n is A_n $Y is\ f(X_1, ..., X_n)$ ——— $Y\ is\ B$	$\mu_B(v) =$ $sup_{u_i}(\bigwedge_{i=1...n}(\mu_A(u_i)))$ $s.t : v = f(u_1, ...u_n)$
General Extension Principle	$g(X_1, . . ., X_n)\ isA$ $Y\ is\ f(X_1, . . . , X_n)$ ——— $Yis?B$	$\mu_B(v) =$ $\sup_{u_i}((\mu_A(g(u_1, ..., u_n))$ $s.t : v = f(u_1, . . ., u_n)$
Compositional Rule of Inference	$XisA$ $(X, Y)isB$ ——— $YisC$	$\mu_C(v) =$ $sup_u((\mu_A(u)\hat{}\mu_B(u, v)))$
FuzzyGraph Interpolation	$\sum_{i=1}^{n} if\ Xis\ A, then\ Y\ is\ B_i$ $XisA$ ——— $YisB$	$\mu_B(v) = \sup_i(m_i \wedge B_i)$ $m_i = \sup_u(\mu_A(u)\hat{}\mu_{A_i}(u))$
Fyzzy Syll-ogism	$Q_1 A's\ areB's$ $Q_2 B's\ areC's$ ——— $Q_3 A's\ areC's$	$\mu_{Q_3}(z) =$ $\sup_{w_1, w_2}(\mu_{Q_1}(w_1)\hat{}\mu_{Q_2}(w_2)))$ $s{:}t : z = w_1 \times w_2$
Fyzzy Probability	$(X$ is $A)\ ispP_1$ ——— $(X$ is $B)\ isp\ P_2$	$\mu P_2(v) =$ $\sup_g(\int_U \mu_A(u)g(u)d(u))$ $s{:}t : v = \int_U mu_B(u)g(u)$ $d(u),$ $\int_U g(u)d(u) = 1$ g = PDF of X

Figure 1 shows the overall view of a CW question answering system. The input to the system is a domain knowledge as well as a set of queries, both expressed in natural language. A CW question answering system composed of three main modules, namely, the translation module, the inference engine, and the retranslation module.

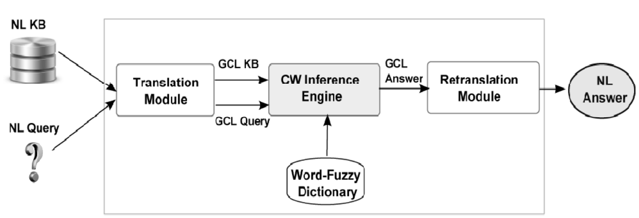


Figure 1: Overall Scheme of a CW Question Answering System

The translation module converts the natural language propositions to GCL expressions. It must be emphasized, however, that an inclusive translation from natural language to GCL is not feasible. The laws of the syntax and semantics of natural language are much more complicated to be grasped by GCL. Natural language statements are context-dependent and contain non-truth functional connectives, such as: "hence", "because", "so that", "conclusively", etc., which cannot be translated into truth-conditional languages such as GCL. As a result, the proper goal in this area would be to determine a subset of natural language expressions with a restricted grammar and semantics which is convertible to GCL and develop an automated translation tool for this subset. The research in this area is still pre-mature and requires a deep knowledge of computational linguistics [4].

The inference engine performs reasoning on a set of given generalized constraints via the inference rules that reside in the deduction database. In order to carry out the computational part of the rules, the inference engine needs to know the fuzzy subsets which correspond to various linguistic terms of a linguistic variable. This information is stored in the wordfuzzy dictionary which contains all the linguistic variables in the problem domain as well as their linguistic term and the fuzzy subsets associated with each linguistic term. The main task of the inference engine is to systematically match the domain knowledge with the deduction rules to draw an answer to a given query.

The rules in the deduction database do not provide a linguistic label for the value of the constrained variable in their right-hand-side, but they merely compute the fuzzy subsets that represent such value. Consequently, the inference engine returns the answer to a query in terms of fuzzy sets. However, in some application, it might be desired to specify the output in terms of linguistic terms rather than fuzzy subsets. Thus a retranslation element is needed to assign the most appropriate linguistic term from the domain vocabulary to each fuzzy subset obtained from the inference engine. The main issue in the retranslation process is to minimize the loss of information due to the linguistic approximation. There have been many articles in the literature that focused on minimizing such loss [5], [2], [11], [12], [22]. A number of criteria for evaluating a retranslation method is proposed in [16].

The focus of this paper is to design the inference engine of a CW QA system. There have been very few studies in the literature regarding a CW inference engine [8], [17], [10]. These studies either did not provide a systematic approach for applying the inference rules or focused on a limited number of rules and did not consider advance CW inference patterns including fuzzy probability and fuzzy syllogism. The methodology, presented in this paper, extracts and organizes the knowledge related to the query in a tree structure, called a Constraint Propagation Tree (CPT). An evaluation algorithm then traverses the tree and propagates the constraints from this set to the query, while combining the different answers obtained for each node. CPT also allows one to identify the missing knowledge when the information in the knowledge base is not enough for providing an answer. We illustrate the methodology by an example and discuss its implementation. In particular, we classify and formulate the knowledge related to the query into three canonical forms and show how this classification can subsume a potentially very large deduction database with merely primary deduction rules, namely, the extension principle and compositional rule of inference.

II. A CW DEDUCTION METHODOLOGY

The deduction methodology takes a knowledge base and a set of queries and makes a sequence of inferences to obtain a direct answer to the query. As the first step, we assume that the knowledge base and the query are translated, manually or automatically, into GCL. The query posed to the system may be of various types. Generally a query can be viewed as seeking a value for one or more constrained variables, i.e., the query is of the general form: X is ?R, where X is a constrained variable and the goal is to find the value of ?R. This view of the query includes a wide range of factual questions, list questions, and truth, and probability qualified questions. Few examples of different query types that can be represented in this form are listed below:

- Factual questions: are questions that have an objective answer, e.g.,
  - "how tall is John?" → "height(John) is ? "
  - "how far are Chicago and St.Louis?" → "distance(Chicago, St.Louis) is ?"
  - "How many women with breast cancer are obese? → $count_x$(weight(x) is obese| hasbc(x)^ woman(x)) is ?
- List questions: are questions that can have more than one answer and each answer is associated with a truth degree. This type of query may be represented by a veristic constraint in GCL [15]. In this case, the constraint variable is disjunctive, i.e., that is it can take more than one value simultaneously , e.g.,
  - "what are the biggest countries in the world?" → "biggest-countries isv ?". The answer to this question may be: "{Russia/1, Canada/1, China/1, United States/.9, Brazil/.8, Australia/.7}"
  - "Who are John's Sisters" → "sisterof(John) isv ?".
- Truth qualified questions: are questions that ask about the degree of truth of a proposition. The answer to this type of question determines how the hypothesis of the question is supported by data in knowledge base. Again the verity constraint can

be used to represent such queries; however, this time the constrained variable is itself a generalized constraint, e.g.,

– "Is it true that unemployment rate is about 9% in the united states" → "(unemployment-rate(US) is about-9%) isv ?"

Our deduction methodology instantiates the query variables in two phases: first the information relevant to the query is extracted and organized in a CPT. Next, the tree is evaluated to find the value of the query variable while combining different values obtained. There are two types of relevancy: direct and indirect. Direct relevancy can be assessed by pattern matching while indirect relevancy requires reasoning and deduction on knowledge base. For example if the query is Q: "price(gas) is ?", and the knowledge base. contains two propositions: P_1: "relation(price(gas), production(oil)) is linear", and P_2:"production(oil) is low", then P_1 is directly and P_2 is indirectly relevant to the query. Formally a proposition P is directly relevant to the query Q if it satisfies one the following conditions:

- P contains the constraint variable and the object variable of Q. For example, the proposition P: "relation (price (gas), production(oil)) is linear", is directly related to the query Q: "price(gas) is ?", because it contains the constraint variable of Q :"price", as well as its object variable:"gas".
- P contains the constraint variable of Q with a generic object variable. For example the propositon P: "if Age(x) is young then risk(BreastCancer(x)) is low", is directly related to Q: "risk(BreastCancer(Mary)) is ?", as the generic object variable, x, can be instantiated to "Mary".

The constraint propagation tree applies the protoformal deduction rules in a hierarchical way to extract the propositions that are directly or indirectly relevant to the query and determines this relation. The root node in CPT represents the input query and the intermediate nodes are sub queries. Each node is connected to its children via a protoform rule, where the parent node represents the consequent and the children nodes represent the premises of the rule. In this perspective, CPT may be viewed as the CW version of a classical and-or tree. However, in a classical and-or tree, the only rule of inference is modus ponens and the only information carried by the parent node about its children is whether they are combined in a conjunctive or a disjunctive manner. But in CW various rules of inference may be applied to the knowledge base and the instantiation of variables include fuzzy computation; hence, a parent node must include information about the type of deduction rules that should be applied to its children. A node in CPT is represented by a quadruple: (N;GC;E), where:

- N is an integer, denoting the node number.
- GC is a generalized constraint with zero or more uninstantiated variables. e.g., "Age(Mary) is ?R" or "" if Age(x) is over 40 then risk(breastCancer(x)) is high".
- E indicates the type of deduction rules that are applied to the immediate children of this node. $E = f(r;N)$, where r is a protoformal rule that instantiates the variables of the current node and N is the set of node numbers of a group of immediate child nodes that form the premises of r. For example let us assume that a node i has children $\{j, k, m, n\}$ where nodes $\{j,k\}$ and $\{m,n\}$ are connected to node i by rules a and b, respectively. In this case $E = \{(a, \{j, k\}), (b, \{m, n\})\}$, indicating that the application of rule a to nodes j and k, as well as rule b to nodes m and n, gives two alternatives for instantiating the variable in node i. Depending on the type of the constrained variable in node i, its final value may be the conjunction or disjunction of the values obtained by these alternatives.

Algorithm 1 shows the abstract procedure of generating a CPT in response to a given query. The Algorithm takes three inputs: A query, a knowledge base, and a deduction database and recursively generates the CPT. A query is a generalized constraint of the form "X is ?R", where X is a constrained variable and the goal is to instantiate ?R. A knowledge base consists of a set of propositions in form of generalized constraints and the deduction database includes a set of CW inference rules. Lines 1-4 initialize the root node to the query and set the current node to the root node. Lines 5-7 show the base case of recursion where all variables of the current node are instantiated. Lines 10-15 extract, from the knowledge base, the set of propositions that are directly relevant to the current node. Such set is called DRS. Lines 16-18 search DRS for a proposition that exactly matches the generalized constraint of the current node. If such proposition is found, the variables of the current node are instantiated accordingly. This is the case where the answer to the query is explicitly stored in the knowledge base. For example, if the query is: *Q*: "Age(Mary) is ?*R*", and the knowledge base contains the proposition "Age(Mary) is middle-age", then *R* is simply instantiated with the fuzzy subset that represents the linguistic value "middle-age". If the answer to the current node is not explicitly stored in the knowledge base, the algorithm continues its search to extracts the implicit relevant knowledge in KB. In lines 20-27, the algorithm looks for a rule in the deduction database whose consequent matches the query and at least one of its premises matches a proposition in DRS. If such rule was found, then, for each premise of the rule, the tree is searched to find a node corresponding to that premise. If such node was found, then it is reused, otherwis a new child node is created (lines 28-38) . The child nodes are considered as new sub-queries and are expanded in case they contain an un-instantiated variable (lines 39-40). The same process is performed for each child node until no new node is generated. Note that the nodes are generated in a depth-firstsearch manner.

Once CPT is generated, the answer to the query may be found by applying the inference rules related to each node. The second phase of deduction is to propagate the constraints from the bottom of CPT to the top while combining different constraints obtained for each node. The propagation and combination algorithm is straightforward. It starts with the nodes in the level before the last level and applies the related inference rules to the appropriate group of children to obtain a constraint for their parent node. Such constraint is, in general, a fuzzy subset on the domain of the constrained variable of the node. If more than one constraint is obtained for a node, then the node variable should be instantiated to the combination of these constraints. If the node variable is possibilistic, then it will be instantiated to the conjunction of the individual constraints, whereas if it is a verisitic variable it can take more than one value, and hence will be instantiated to the disjunction of the individual constraints. For instance, assume that we are interested to know how heavy John is, and we inferred two pieces of information from KB about the possibilistic constrained variable "weight(John)": (1) "weight(John) is not very heavy", and (2) "weight(John) is a little heavier than 50 kilos". Since John can only have one value Combining these data, we can conclude that "weight(John) is not very heavy ^ a little heavier than 50 kilos". As an example of veristic constraint combination, suppose that we inferred, from KB, two fuzzy sets for the countries that John would like to visit: (1) {Egypt/1, Spain/0.8, Canada/0.5}, and (2) {China/0.6, Jordan/0.3}; the variable "country(likedby(John)" can assume more than one value, hence: "(likeby(John)) isv {Egypt/1, Spain/0.8, Canada/0.5} _ {China/0.6, Jordan/0.3}".

The fuzzy set obtained from applying the protoform rules should be normalized before being propagated to its upper level. After instantiation, the values found for the intermediate nodes may be stored and reused for future queries, provided that the

Algorithm 1 CPT Generation Algorithm

```
Input: Q:query, DDB:deduction-database, KB: knowledge base
 1: Create a new node for root
 2: count=0;
 3: root.GC ← Q root.N ← count root.E ← {};
 4: node ← root
 5: if node.GC is fully instantiated then
 6:    return
 7: end if
 8: x ← the constraint variable of node.GC
 9: DRS←{}
10: for all GeneralizedConstraint P in KB do
11:    xp ← the constrained variable of P
12:    if (xp contains x) & ((object-variable of xp== object-variable
       of x) or (object-variable of xp is generic)) then
13:       DRS=DRS ∪ {P}
14:    end if
15: end for
16: for all generalizedConstraint P in DRS do
17:    if node.GC matches P then
18:       instantiate node.GC with P
19:    else
20:       for all inferenceRule r in DDB do
21:          C= r.consequent
22:          for all GeneralizedConstraint A in r.premises do
23:             if A matches P & C matches node.GC then
24:                isMatched ← true
25:                break
26:             end if
27:          end for
28:          if isMatched then
29:             parent ← node
30:             childNumbers ← {}
31:             for all GeneralizedConstraint A in r.premises do
32:                search the tree for a node whose GC is A, call it
                   succ
33:                if succ==NULL then
34:                   create a new child node and store it in succ
35:                   count ← count + 1
36:                   succ.GC ← A, succ.E ← {}, succ.N ← count
37:                end if
38:                childNumbers ← childNumbers ∪ {succ.N}
39:                node ← succ
40:                Goto Step 5
41:             end for
42:             parent.E ← parent.E ∪ {(r,childNumbers)}
43:          end if
44:       end for
45:    end if
46: end for
```

information about that variable has not been changed in the knowledge base. The result obtained from the constraint propagation phase is in form of a fuzzy subset on the domain of the query variable. This fuzzy subset can be converted into a numerical value, using a common defuzzification method [13]. Or if a linguistic answer is required, it can be retranslated back to the most appropriate word in the domain of the query variable by using a similarity measure [1].

III. AN ILLUSTRATIVE EXAMPLE

To evaluate our methodology, we applied it to a real world example taken from a web article about breast cancer risk assessment. Suppose that the knowledge base consists of the following information:

Algorithm 2 CPT Evaluation Algorithm

```
Input: Input:CPT
h ← height of CPT
d ← h-1
for all nodes n at depth d of CPT do
    x ← the constrained variable of n.GC
    value ← null;
    for all elements e in n.E do
        fs ← The fuzzy set resulted from applying the inference
        rule in e to the corresponding child node
        normalize fs
        if x is verisite constrained variable then
            value ← value ∨ fs
        else
            value ← value ∧ fs
        end if
    end for
    d ← d − 1
end for
```

> *The average chance that a woman being diagnosed by breast cancer is a function of age. From age 30 through age 39, it is about 0.4%; from age 40 through age 49, it is about 1.5%; from age 50 through age 59, it is about 2.5%, and from age 60 through age 69 it is about 3.5%. There are also some other factors that affect the average risk of breast cancer. Alcohol increases the average risk of breast cancer significantly; pregnancy in the age of 30 or before reduces the average risk of breast cancer by about 3%, and in older women being overweight can slightly increase the average risk of breast cancer.*

Suppose also that we have the following facts about the individual Mary in our knowledge base:

> Mary has a son who is about 20. She gave birth to her son when she was in her 20s. Mary is few years younger than Ann who is in her mid 50. Mary consumes about 1400 to 2000 calories a day. And she drinks moderately.

As commonsense knowledge, we also know that:

> Overeating causes a person to be overweight and the age of a mother is equal to the age of her son plus the age that she gave birth to her son.

Given the above information we are interested to know Mary's chances of developing a breast cancer. Suppose that the query and the knowledge base are translated to GCL as follows:

Query: riskbc(Mary) is ?

Knowledge base:

(P1) if age(x) is in 30s then average-riskbc(x) is about 0.4% +
if age(x) is in 40s then average-riskbc(x) is about 1.5% +
if age(x) is in 50s then average-riskbc(x) is about 2.5% +
if age(x) is in 60s then average-riskbc(x) is about 3.5%

(P2) if dirnkHabit(x) is regularly then alcoholFactor(riskbc(x)) is significant

(P3) if age(pregnancy(x)) is about 30 or before then pregFactor(riskbc(x)) is about 3%

(P4) if age(x) is old and weight(x) is overweight then weightFactor(riskbc(x)) is slightly

(P5) riskbc(x) is average-riskbc(x) + alcoholFactor(riskbc(x)) + weightFactor(riskbc(x)) - pregFactor(riskbc(x))

(P6) age(son-of(Mary)) is about 20

(P7) age(Mary) is Age(Ann) - few years

(P8) age(Ann) is mid-50

(P9) age(pregnancy(Mary)) is in 20s

(P10) eatingHabit(Mary) is about 1400 to 2000 calories per day

(P11) if eatingHabit(x) is overeat then weight(x) is overweight

(P12) age(x) is age(son(x)) +age(pregnancy(x))

(P13) drinkHabit(Mary) is moderate

The CPT of this example is shown in figure 2. Rules "EP" and "FG" stand for extension principle and fuzzy graph interpolation rules, respectively (see table I). The root node indicates the query and the variable that needs to be instantiated is "risk(bc(Mary))". The propositions that are directly related to this variable are P1-P5, but only P5 may instantiate the query variable via the extension principle (EP). Hence, the root node is expanded and new child nodes are created to obtain the values of " average-riskbc(Mary)", "alcoholFactor(riskbc(Mary))", "weightFactor (riskbc(Mary))", and "pregFactor(riskbc(Mary))". Note that the generic object variable "*x*" in P5 is substituted with "Mary". Node 2 is fully instantiated and is not further expanded. the propositions that are directly related to node 3 are P1 and P5. But only P1 may instantiate the variable "average-riskbc(Mary)" in node 3 via the fuzzy graph interpolation rule. Thus nodes 4 and 5 are created for the premises of this rule. Node 4 is fully instantiated but the value "age(Mary)" in node 5 needs to be determined. The propositions that are directly related to node 5 are P1, P4, P7, and P12, but only P7 and P12 lead to instantiation of the node variable. P7 relates the variable "age(Mary" to "age(Ann") via the extension principle and results in generation of nodes 6 and 7. P12 relates "age(Mary)" to "age(son(Mary))" and "age(pregnancy(Mary))" and yields nodes 8,9, and 10. The rest of the tree is generated similarly.

Before evaluating the tree A word-fuzzy dictionary must be provided which defines the linguistic variables in the problem domain as well as their linguistic terms and the fuzzy subsets associated with each term. A word-fuzzy dictionary for the above example is shown in table II. For simplicity, all the membership functions are defined linear. The evaluation of the tree starts with instantiating nodes 5 and 19 . The value of "Age(Mary)" in node 5 is obtained by the application of extension principle twice; the first time, function f is the fuzzy subtraction of the values of "age(Ann)", and "few-years', and the second time, function f is the fuzzy addition of the values of "age(son(Mary))" and "age(pregnancy(Mary))". Since "age(Mary)" is a possibilistic variable, node 5 is instantiated to the conjunction of the result of the two values. Node 19 is instantiated by applying the fuzzygraph interpolation to nodes 20 and 21. The next nodes that are instantiated are 3, 11, 14, and 16 whose values are obtained by applying the fuzzygraph rule to the related child nodes. Finally, the query variable is instantiated by the application of extension principle to nodes 2, 3, 11,14,and 16. The answer to the query is obtained as a fuzzy subset over the domain of "riskbc" as shown in figure 3. If required, this answer can be defuzzified to provide a single value. Using the centroid defuzzification method the answer to the query is: "Risk(bc(Mary)) is 3.2%".

One advantage of organizing the knowledge in form of a CPT is that it can serve to seek additional information when the information in knowledge base is not enough to answer a query. For instance, in the above example, if we exclude the proposition: "drinkHabit(Mary) is moderate" from the knowledge base, then the value of "alcoholFactor(riskbc(Mary))", node 11, remains unknown and the root node cannot be instantiated. In this case, CPT can help to identify the missing knowledge and establish a dialog with user to obtain this knowledge. A node in CPT is considered a missing knowledge if: (1) it has an un-instantiated variable,(2) the variable cannot be instantiated by the set of directly related propositions in KB (DRS), and

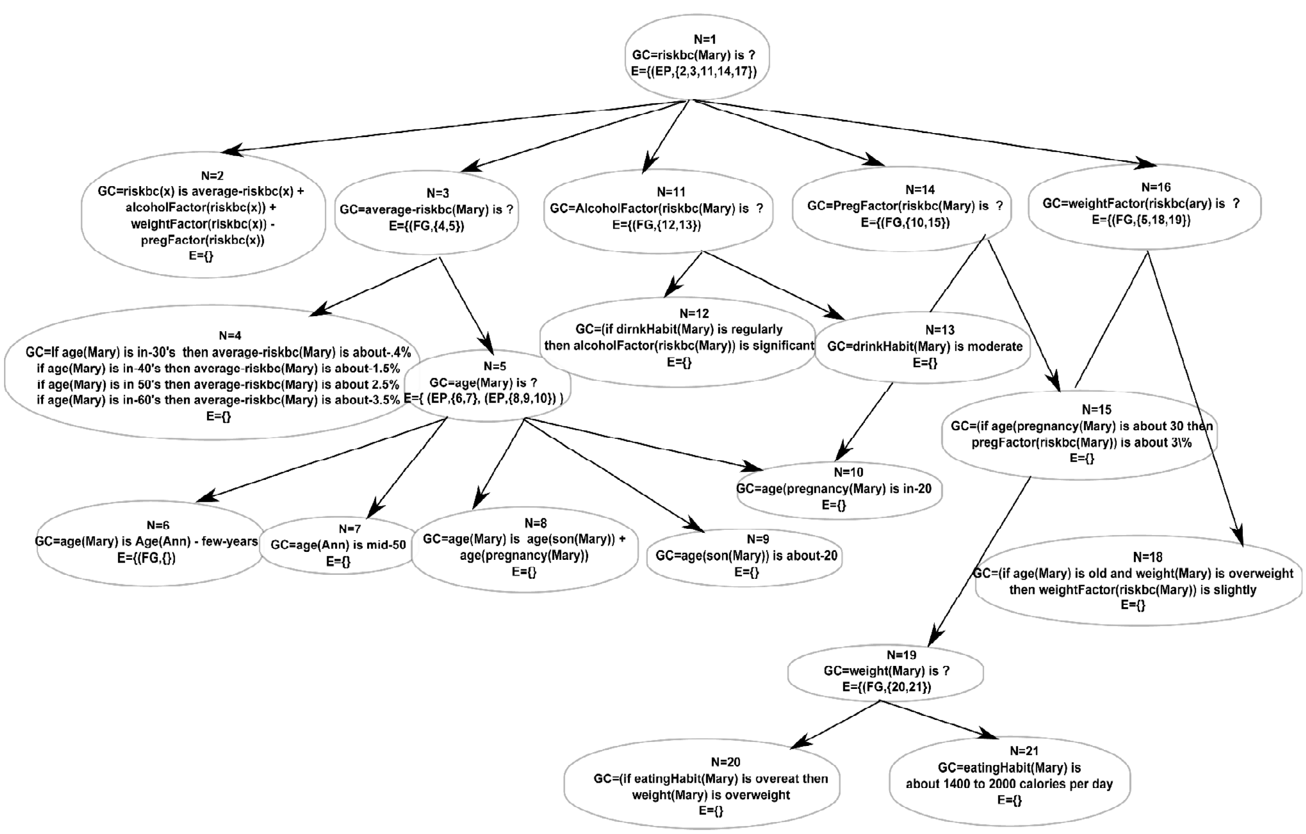


Figure 2: The CPT Generated in Response to the Query: "riskbc(Mary) is ?"

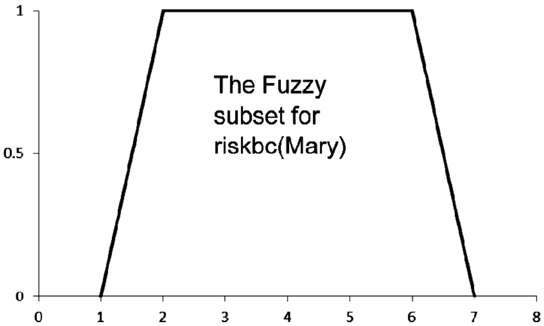


Figure 3: The Fuzzy Subset Obtained for Riskbc(Mary) after the Evaluation of CPT

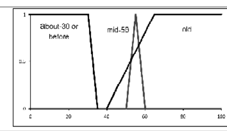
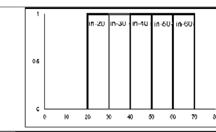
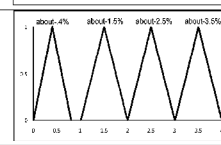
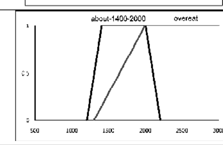
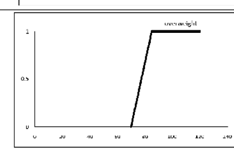
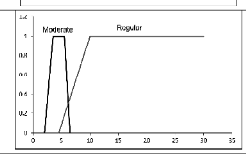
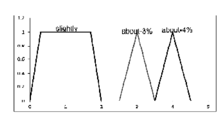
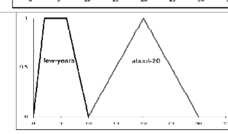
(3) the node cannot be expanded. The missing knowledge may be required or not required for answering a query. A required missing knowledge must be provided to find an answer to the query, for example, if we exclude the proposition "drinkHabit(Mary) is moderate" then node 13 becomes a required missing knowledge. A non-required knowledge is not necessary for answering the query; however, providing this knowledge may improve the quality of the answer. For example if the proposition "Age(Ann) is mid-50" is removed from the knowledge base, node 7 becomes a non-required missing knowledge as the root node can still be instantiated. However, knowing the value of "age(Ann)" will give us additional knowledge about "age(Mary)" and will lead to a better estimate for "average(risk(bc(Mary))", and "risk(bc(Mary))", consequently.

IV. A CANONICAL DEDUCTION FORMULATION

The list of protoformal deduction rules in the deduction database is not comprehensive [21]. The number of syntactic forms that a proposition can take in natural language and GCL, accordingly, is very large. Thus it is very inefficient (if not impossible) to define a comprehensive deduction database that can be matched against all syntactic extensions in GCL. For instance, suppose that in the previous example, we replace the piece of commonsense knowledge: "overeating causes being overweight" with a more realistic one: "usually overeating causes being overweight". This proposition can be represented as a generalized constraint: " if eatingHabit(x) is overeat then weight(x) isu overweight". With this modification, the fuzzygraph interpolation rule cannot be applied to obtain a value for the linguistic variable "weight(Mary)" in figure 2. Hence a new version of the interpolation rule that includes usuality constraint should be developed and added to the deduction database. The symbolic part of this rule can be stated as follows:

$$\begin{array}{l} \sum_{i=1}^{n} if\ X\ is\ A_i\ then\ Y\ isu\ B_i \\ X\ is\ A \\ \hline Y\ isr\ ?\ B \end{array}$$

Table II
A Word-fuzzy Dictionary: The Linguistic Variables And Their Associated Linguistic Terms

Linguistic Variable	Linguistic Terms	Linguistic Variable	Linguistic Terms
age		age	
riskbc		eatingHabit (calories/day)	
weight(Kilos)		drinkHabit	
riskFactor		Age	

As another example of the data that cannot be matched with the currently developed deduction rules, suppose that we are interested to estimate the total university budget for year 2010 knowing that due to the recession in economy there might be a slight cut in the budget for 2010 compared to 2009. We also know that the university budget for 2009 is about $140 million. The data and the query in this case can be converted to generalized constraints as:

$$\begin{array}{l} \text{Budget(2009) is about-\$140m} \\ \text{(Budget(2010) is Budget(2009)slightAmount) isp likely} \\ \hline \text{Budget(2010) is ?} \end{array}$$

This data matches with the following protoformal rule which has not yet been developed in the deduction database:

$$\begin{array}{l} X \text{ is } A \\ (y \text{ is } f\,(X)) \text{ isp } P \\ \hline Y \text{ isp } ?B \end{array}$$

Similarly, one can think of a large number of inference patterns that needs be developed and added to the deduction database based on the various forms of a GCL expression. However, a large deduction database leads to a poor efficiency in implementation of a CPT as each proposition related to the query needs to be checked against all deduction rules in order to find a match. Therefore, it is very helpful to define a set of canonical deduction forms that can subsume the large list of protoformal deduction rules corresponding to various inference patterns in GCL. In sequel, we classify the knowledge related to the query in to three canonical forms and show how the primary rules of inference, namely the extension principle and the compositional rule of inference, can be used to instantiate the query variable in all of these cases. Before proceeding, the following notations are needed:

- Y: denotes the query constraint variable, i.e., the variable whose values are constrained by the answer.
- $X_1, X_2, \ldots, X_n$: are fuzzy variables.
- $U_1, U_2, \ldots, U_n, V$: are universes of discourse of $X_1, X_2, \ldots, X_n$, and Y, respectively.
- *A*, *B*, *C*, *D*: are fuzzy values.
- f, g: are crisp functions.
- $(X_1, \ldots, X_n)$: is a relation on the fuzzy variables $X_1, X_2, \ldots, Xn$.

The propositions in the knowledge base that are directly related to the query: "*Y* is ?", take one of the following canonical forms:

Canonical form i: Y is A

This form of data explicitly instantiates the query variable without performing any deduction. For example:

Query: price(oil) is ?

Data: price(oil) is about $3 a gallon

The appearance of canonical form I in the knowledge base instantiates the query variable but does not cause the query node to expand in CPT.

Canonical form ii: *Y* is *f* ($X_1, \ldots, X_n$)

This form of data defines the query variable as a function of some other variables. Therefore to instantiate the query variable we first need to find the value of those variables. Consequently, the appearance of canonical form ii in the knowledge base causes the query node to expand to obtain the values of the variables $X_1, X_2, \ldots, X_n$, as shown in figure 4. After instantiating the variables $X_1, X_2, \ldots, X_n$, the extension principle is used to compute the value of Y as follows:

$$\mu_B(v) = \sup_{u_i}\left(\bigwedge_{i=1\ldots n} (\mu_A(u_i))\right),\ s,t: v = f(u_1,\ldots,u_n)$$

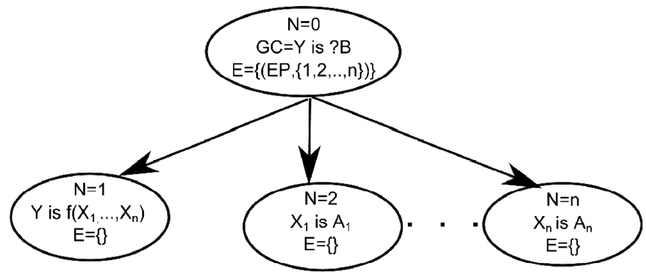


Figure 4: CPT Corresponding to Canonical form ii

Canonical form ii has another variant and that is when the knowledge base does not contain the value of the variables $X_1, \ldots, X_n$ but, instead, it contains the value of a function of these variables $g(X_1, \ldots, X_n)$. The CPT corresponding to this canonical form is depicted in figure 5. In this case, the general extension principle (GEP, table I) is used to find the value of *Y* as follows:

$$\mu_B(v) = \sup_{u_i}((\mu_A(g(u_1,\ldots,u_n)),\ s.t: v = f(u_1,\ldots,u_n)$$

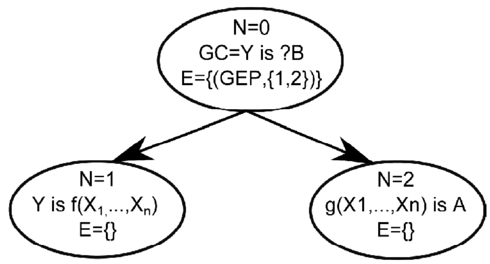


Figure 5: CPT Corresponding to a Variant of Canonical form ii

Canonical form iii: (*Y*, $X_1, \ldots, X_n$) is *C*

In this form, the data related to the query is a fuzzy relation on the query variable and a set of other variables. The appearance of canonical form iii in the knowledge base causes the query node to expand to obtain the value of the variables $X_1, \ldots, X_n$, as portrayed in figure 6. After the instantiation of the variables $X_1, \ldots, X_n$, the compositional rule of inference (CRI) is used to derive the value of the query variable, Y, as follows:

$$\mu_C(v) = \sup_{u_1,\ldots,u_n}(\bigwedge_{i=1\ldots n} \mu_{A_i}(u_i) \wedge \mu_B(u_1,\ldots,u_n,v)))$$

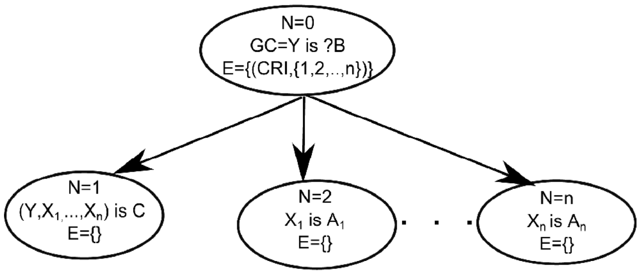


Figure 6: CPT Corresponding to the Canonical form iii

Classifying the knowledge related to the query into the canonical forms i, ii, and iii allows subsuming a (potentially) large list of deduction rules with the three primary rules, namely, EP, GEP, and CRI. This facilitates the implementation of CPT and avoids developing new protoformal rules for various syntactical inference patterns in GCL. Nevertheless, a preprocessing phase is required to reformulate the knowledge related to each query in to canonical forms. In what follows we shall show how we can reformulate and derive various GCL inference patterns by using the canonical forms. First, a set of rewriting rules for GCL expressions are listed.

Rewriting formulas with fuzzygraph constraint

The formula $\sum_{i=1}^{n}$ if X is A_i then Y is B_i imposes a constraint on the fuzzy relation (X, Y) and may be reformulated as follows:

$$\sum_{i=1}^{n} if\ X\ is\ A_i\ then\ Y\ is\ B_i \rightarrow (X,Y) is\ B$$

where $\mu_B(v) = sup_i(\mu_{A_i}(u), \mu_{B_i}(v))$.

In general, the formula $\sum_{i=1}^{n} if\ f_i(X) is\ A_i\ then\ g_i(Y) is\ B_i$, where f and g are crisp function, imposes a constraint on the fuzzy relation (X, Y) and is reformulated as follows:

$$\sum_{i=1}^{n} if\ f_i(X) is\ A_i\ then\ g_i(Y) is\ B_i \rightarrow (X,Y) is\ B$$

where $\mu_B(v) = \sup_i(\mu_{A_i}(f(u)); \mu_{B_i}(g(v)))$

Rewriting formulas with fuzzy quantifiers

The formula Q As are Bs, where Q is a relative fuzzy quantifier [6], [7], such as: "many", "most", "some",etc, and A and B are fuzzy values, imposes a constraint on the relative cardinality of fuzzy sets A and B. The relative cardinality denotes the proportion of the elements of fuzzy set A that are also in B and is defined as: card($B|A$)= $\frac{card(A \wedge B)}{card(A)}$. Various methods for measuring the cardinality of a fuzzy set exists in the literature, such as: sigma-count, FG-count [18], FE-count, and ordered weighted average (OWA)[14]. The classical approach for measuring the cardinality of a fuzzy set, is Zadeh's sigma-count method, where: card(A) = $\sum_{u \in U} (\mu_A)(u)$. The quantified formula may be rewritten as follows:

$$QAs\ are\ Bs \rightarrow card(A|B)\ isQ$$

Rewriting formulas with fuzzy probabilities

The formula (X is A) isp$P$, where P is a fuzzy probability, such as: "likely", "probably", "certainly", etc, imposes a constraint on the probability distribution, r, of variable X and can be represented as a function of such distribution:

$$(X\ is\ A) is\ p\ P \rightarrow f(r) = \int_U \mu_A(u) r(u) d(u) isP$$

Rewriting formulas with fuzzy usuality constraint

The formula X isu A is semantically equivalent to (X is A) *isp usually* and hence it can be rewritten as:

$$(X\ isuA) \rightarrow f(r) = \int_U \mu_A(u) r(u) d(u) is\ usally$$

Using the above rewriting rules, many inference patterns are automatically reduced to a canonical deduction form. Few examples of such reduction are listed below. The first three examples are the fuzzygraph interpolation, fuzzy probability, and fuzzy syllogism rules, as listed in table I. We show how the semantical part of such rules may be derived by rewriting the data in canonical forms. The last example shows a new inference pattern which is developed and reduced into a canonical form.

- Fuzzy graph interpolation rule: using the previous rewriting rules, the fuzzygraph interpolation rule is reduced to the following deduction:

$$\begin{array}{l} X \text{ is } A \\ \underline{(X, Y) \text{ is} C} \\ Y \text{ is } ?B \end{array}$$

Where $\mu_C(v)=\sup_i(\mu_{A_i}(u) \wedge \mu_{B_i}(v))$. This deduction complies with the canonical deduction form iii, and hence the value of B is computed using the compositional rule of inference:

$$\mu_B(v)=\sup_u(\mu_A(u),\mu_C(u,v)=\sup_u(\mu_A(u)\wedge\sup_i(\mu_{A_i}(u)\wedge\mu_{B_i}(v))$$

The sup operation is distributive with respect to $\wedge$, hence:

$$\mu_B(v)=\sup_u(\mu_A(u))\wedge\sup_i(\sup_u(\mu_{A_i}(u))\wedge\mu_{B_i}(v))$$

$$=\sup_i(\sup_u(\mu_A(u)\wedge\mu_{A_i}(u))\wedge\mu_{B_i}(u))$$

which results in the same value as the semantic part of the fuzzy graph interpolation rule in table I.

- *fuzzy probability* rule: using the previous rewriting rules, the fuzzy probability rule is reduced to the following deduction:

$$\begin{array}{l} \int_U \mu_A(u)r(u)d(u) is\, P_1 \\ Y\, is \int_U \mu_B(u)r(u)d(u) \\ \hline Y\, is\, P_2 \end{array}$$

This deduction fits the canonical deduction form ii and the value of P_2 is computed using the general extension principle:

$$\mu_{P_2}(z)=\sup_r(\mu_{P_1}(\int_U \mu_A(u)r(u)d(u)))s.t.z=\int_U \mu_B(u)r(u)d(u)$$

Which coincides with the semantic part of the fuzzy probability rule in table I.

- *fuzzy syllogism rule:* the fuzzy syllogism rule is reduced to the following deduction:

$$\begin{array}{l} \text{card}(B|A) \text{ is } Q_1 \\ \text{card}(C|A\wedge B) \text{ is} Q_2 \\ \hline \text{card}(B\wedge C|A) \text{ is } ?Q_3 \end{array}$$

At first glance, this deduction does not seem to match any canonical deduction form; however, based on the definition of the relative cardinality, it is clear that : card($B \wedge C|A$) = card($B|A$)card($C|A \wedge B$), adding this additional implicit information to the premises of the above deduction, we have:

card($B|A$) isQ_1

card($C|A\wedge B$) isQ_2

card($B\wedge C|A$) is card($B|A$)card($C|A\wedge B$)

card($B\wedge C|A$) is ?Q_3

This deduction fits the canonical deduction form ii, and the value of Q_3 is calculated as follows:

$Q_3(z) = \sup u_1, u_2(Q_1(w_1)\wedge Q_2(w_2)),\ s.t.z = w_1\times w_2$

Which coincides with the semantic part of the fuzzy syllogism rule in table I.

- usuality-qualified interpolation rule: previously we showed that if we change the proposition "if eatinghabit(x) is overeat then weight(x) is overweight" to the usuality constrained proposition: "if eating-habit(x) is overeat then weight(x) isu overweight" in the CPT of figure 2, then we need to adopt the fuzzy interpolation rule to include usuality constraint, that is, to develope the following protoformal rule:

$$\begin{array}{l} X \text{ is} A \\ \sum_i \text{ if } X \text{ is} A_i \text{ then} Y \text{ isu} B_i \\ \hline Y \text{ isr } ?B \end{array}$$

As mentioned before the usuality formula, "Y isu B_i" imposes a constraint on the probability distribution of Y.

Rewriting this formula we get the following deduction:

$$\begin{array}{l} X \text{ is} A \\ \sum_i \text{ if } X \text{ is} A_i \text{ then } f_i(r) \text{ is usually} \\ \hline Y \text{ isr } ?B \end{array}$$

where r is the probability distribution function of Y, and $f_i(r) = \int_v \mu_{B_i}(v)r(v)d(v)$. After rewriting the formula with fuzzy graph, we have:

$$\begin{array}{l} X \text{ is} A \\ (X, r) \text{ is} C \\ \hline r \text{ is } ?B \end{array}$$

Where:

$$\mu_C(u,r)=\sup_i(\mu_{A_i}(u)\wedge\mu_{usually}(\int_v r(v)d(v)\mu_{B_i}(v))$$

This deduction coincides with canonical deduction form iii, and the value of B is obtained as follows:

$$\mu_B(r)=\sup_u(\mu_A(u)\wedge\mu_C(u,r))$$

$$=\sup_u(\mu_A(u)\wedge\sup_i(\mu_{A_i}(u)\wedge\mu_{usually}(\int_v r(v)d(v)\mu_{B_i}(v))))$$

$$=\sup_i(\sup_u(\mu_A(u)\wedge\mu_{A_i}(u))\wedge\mu_{usually}(\int_v r(v)d(v)\mu_{B_i}(v)))$$

V. SUMMARY AND FUTURE WORK

Current CW-based question answering systems do not provide a systematic approach for extracting and integrating the information in the knowledge base. In this work we developed an inference methodology for a CW-based question answering system. The core of methodology is to extract the knowledge relevant to the query and organize it in a constraint propagation tree. The answer to the query may be found by evaluating the CPT. organizing the knowledge in a CPT also helps to achieve a more robust answer by identifying the missing information and obtain it via establishing an information seeking dialog with the user.

To facilitate the implementation of a CPT we classified the knowledge related to the query in to three canonical forms and proposed a set of rewriting rules to convert the data related to a query into one of these forms. We showed, by few examples, how other CW protoform rules and inference patterns may be reduced into a canonical form and derived using the extension principle or the compositional rule of inference. Classification of knowledge into the canonical forms reduces a potentially very large deduction database to these primary inference rules.

In order to make the implementation scalable to larger domains, we are planning to develop appropriate off-line techniques to store and update the data from previously generated CPTs in an indexed database to use in later queries. Also to reduce the size of a CPT, The generation algorithm should be modified to stop searching after finding a reasonable answer according to the user's expectations.

References

[1] V. V. Cross and T. A. Sudkamp. *Similarity and Compatibility in Fuzzy Set Theory: Assessment and Applications*. Physica-Verlag GmbH, Heidelberg, Germany, Germany, 2002.

[2] A. Dvoràk. On Linguistic Approximation in the Frame of Fuzzy Logic Deduction. Soft Computing - A Fusion of Foundations, *Methodologies and Applications,* **3**(2): 111–116, 1999.

[3] C. Fellbaum, editor. WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press, 1998.

[4] J. Kacprzyk and S. Zadrozny. Computing with Words is an Implementable Paradigm: Fuzzy Queries, Linguistic Data Summaries, and Natural-language Generation. Fuzzy Systems, *IEEE Transactions on*, **18**(3): 461–472, 2010.

[5] R. Kowalczyk. On Linguistic Approximation of Subnormal Fuzzy Sets. In Fuzzy Information Processing Society - NAFIPS, 1998 Conference of the North American, 329–333, 1998.

[6] Y. Liu and E. E. Kerre. An Overview of Fuzzy Quantifiers. (i). Interpretations. *Fuzzy Sets and Systems,* **95**(1): 1–21, 1998.

[7] Y. Liu and E. E. Kerre. An Overview of Fuzzy Quantifiers. (ii) Reasoning and Applications. *Fuzzy Sets and Systems*, **95**(2): 135–146, 1998.

[8] J. Mendel and D. Wu. Perceptual Computing: Aiding People in Making Subjective Judgments. Wiley-IEEE Press, 2010.

[9] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan. LCC Tools for Question Answering. notebook of the Eleventh Text Retrieval Conference (TREC2002), pages 144–154, 2002.

[10] Z. Qin, M. Thint, and M. M. Sufyan Beg. Deduction Engine Design for pnl-based question answering system. *In Proceedings of the 12th International Fuzzy Systems Association World Congress on Foundations of Fuzzy Logic and Soft Computing*, IFSA '07, pages 253–262, Berlin, Heidelberg, 2007. Springer-Verlag.

[11] T. Whalen and B. Schott. Empirical Comparison of Techniques for Linguistic Approximation. In IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th, **1**, 93–97, 2001.

[12] D. Wu and J. M. Mendel. A Vector Similarity Measure for Linguistic Approximation: Interval type-2 and type-1 fuzzy sets. *Inf. Sci.*, **178**(2): 381–402, 2008.

[13] L. W.V. and K. E.E. Defuzzification: Criteria and Classification. *Fuzzy Sets and Systems*, **108**(2): 159–178, 1999.

[14] R. Yager. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking. *Systems, Man and Cybernetics, IEEE Transactions on*, **18**(1): 183 – 190, jan/feb 1988.

[15] R. Yager. Veristic variables. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, **30**(1): 71–84, 2000.

[16] R. Yager. On the Retranslation Process in Zadeh's Paradigm of Computing with Words. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, **34**(2): 1184–1195, 2004.

[17] R. R. Yager. Knowledge Trees and Protoforms in Question-answering Systems: Special Topic Section on Soft Approaches to Information Retrieval and Information Access on the Web. *J. Am. Soc. Inf. Sci. Technol.*, **57**: 550–563, 2006.

[18] L. A. Zadeh. A Computational Approach to Fuzzy Quantifiers in Natural Languages. Computers & Mathematics with Applications, **9**(1): 149–184, 1983.

[19] L. A. Zadeh. From Computing with Numbers to Computing with Words. *Annals of the New York Academy of Sciences*, **929**(1): 221–252, 2001.

[20] L. A. Zadeh. Toward a Generalized Theory of Uncertainty (gtu): An Outline. *Inf. Sci. Inf. Comput. Sci.*, **172**(1-2): 1–40, 2005.

[21] L. A. Zadeh. From Search Engines to Question Answering Systems U The Problems of World Knowledge, Relevance, Deduction and Precisiation. Elsevier B.V., 2006.

[22] S. Zahan. Neural Approach to Linguistic Approximation of Fuzzy Sets. *In IFSA World Congress and 20th NAFIPS International Conference,* 2001. Joint 9th, pages 2693–2697 **5**, 2001.