UNIT-5

Natural Language Processing

Natural Language Processing (NLP) is the process of computer analysis of input provided in a human language (natural language), and conversion of this input into a useful form of representation.

The field of NLP is primarily concerned with getting computers to perform useful and interesting tasks with human languages. The field of NLP is secondarily concerned with helping us come to a better understanding of human language.

- The input/output of a NLP system can be: –
  written text
  – **speech**
- We will mostly concerned with written text (not speech).
- To process written text, we need:
  – **lexical, syntactic, semantic knowledge about the language** – **discourse information, real world knowledge**
- To process spoken language, we need everything required to process written text, plus the challenges of speech recognition and speech synthesis.

There are two components of NLP.

- **Natural Language Understanding**
  – Mapping the given input in the natural language into a useful representation.
  – Different level of analysis required: ***morphological analysis, syntactic analysis, semantic analysis, discourse analysis***, …
- **Natural Language Generation**
  – Producing output in the natural language from some internal representation.
  – Different level of synthesis required: ***deep planning*** (what to say), ***syntactic generation***
- NL Understanding is much harder than NL Generation. But, still both of them are hard.

The difficulty in NL understanding arises from the following facts:

- Natural language is extremely rich in form and structure, and **very ambiguous**. – How to represent meaning,
  – Which structures map to which meaning structures.
- One input can mean many different things. Ambiguity can be at different levels.

  - Lexical (word level) ambiguity -- different meanings of words
  - Syntactic ambiguity -- different ways to parse the sentence
  - Interpreting partial information -- how to interpret pronouns
  - Contextual information -- context of the sentence may affect the meaning of that sentence.
- Many input can mean the same thing.
- Interaction among components of the input is not clear.

The following language related information are useful in NLP:

- **Phonology** – concerns how words are related to the sounds that realize them.
- **Morphology** – concerns how words are constructed from more basic meaning units called morphemes. A morpheme is the primitive unit of meaning in a language.
- **Syntax** – concerns how can be put together to form correct sentences and determines what structural role each word plays in the sentence and what phrases are subparts of other phrases.
- **Semantics** – concerns what words mean and how these meaning combine in sentences to form sentence meaning. The study of context-independent meaning.
- **Pragmatics** – concerns how sentences are used in different situations and how use affects the interpretation of the sentence.
- **Discourse** – concerns how the immediately preceding sentences affect the interpretation of the next sentence. For example, interpreting pronouns and interpreting the temporal aspects of the information.
- **World Knowledge** – includes general knowledge about the world. What each language user must know about the other's beliefs and goals.

Ambiguity

I made her duck.

- How many different interpretations does this sentence have?
- What are the reasons for the ambiguity?
- The categories of knowledge of language can be thought of as ambiguity resolving components.
- How can each ambiguous piece be resolved?
- Does speech input make the sentence even more ambiguous? – Yes
  – deciding word boundaries
- Some interpretations of : **I made her duck.**

1. I cooked *duck* for her.
2. I cooked *duck* belonging to her.
3. I created a toy duck which she owns.
4. I caused her to quickly lower her head or body.
5. I used magic and turned her into a *duck*.

- duck – morphologically and syntactically ambiguous: noun or verb.
- her – syntactically ambiguous: dative or possessive.
- make – semantically ambiguous: cook or create.
- make – syntactically ambiguous:
  – Transitive – takes a direct object. => 2 – Di-transitive – takes two objects. => 5 – Takes a direct object and a verb. => 4

Ambiguities are resolved using the following methods.

- *models* and *algorithms* are introduced to resolve ambiguities at different levels.
- **part-of-speech tagging** -- Deciding whether duck is verb or noun.
- **word-sense disambiguation** -- Deciding whether make is create or cook.
- **lexical disambiguation** -- Resolution of part-of-speech and word-sense ambiguities are two important kinds of lexical disambiguation.
- **syntactic ambiguity** -- her duck is an example of syntactic ambiguity, and can be addressed by probabilistic parsing.

Models to represent Linguistic Knowledge

- We will use certain formalisms (*models*) to represent the required linguistic knowledge.
- **State Machines** -- FSAs, FSTs, HMMs, ATNs, RTNs
- **Formal Rule Systems** -- Context Free Grammars, Unification Grammars, Probabilistic CFGs.
- **Logic-based Formalisms** -- first order predicate logic, some higher order logic.
- **Models of Uncertainty** -- Bayesian probability theory.

Algorithms to Manipulate Linguistic Knowledge

- We will use *algorithms* to manipulate the models of linguistic knowledge to produce the desired behavior.
- Most of the algorithms we will study are **transducers** and **parsers**.
  – These algorithms construct some structure based on their input.
- Since the language is ambiguous at all levels, these algorithms are never simple processes.
- Categories of most algorithms that will be used can fall into following categories. – state space search
  – dynamic programming

Natural Language Understanding

The steps in natural language understanding are as follows:

Words

Morphological Analysis

Morphologically analyzed words *(another step: POS tagging)*

Syntactic Analysis

Syntactic Structure

Semantic Analysis

Context-independent meaning representation

Discourse Processing

Final meaning representation

Natural Language Generation

The steps in natural language generation are as follows.

Meaning representation

Utterance Planning

Meaning representations for sentences

Sentence Planning and Lexical Choice

Syntactic structures of sentences with lexical choices

Sentence Generation

Morphologically analyzed words

Morphological Generation

Words

Steps in Language Understanding and Generation

Morphological Analysis

- Analyzing words into their linguistic components (morphemes).
- Morphemes are the smallest meaningful units of language.

cars	car+PLU	
giving	give+PROG	
geliyordum	gel+PROG+PAST+1SG	- I was coming

- Ambiguity: More than one alternatives

flies	flyVERB+PROG	
	flyNOUN+PLU	
adam	adam+ACC	- the man (accusative)
	adam+P1SG	- my man
	ada+P1SG+ACC	- my island (accusative)

Parts-of-Speech (POS) Tagging

- Each word has a part-of-speech tag to describe its category.
- Part-of-speech tag of a word is one of major word groups (or its subgroups).
  - **open classes** -- noun, verb, adjective, adverb
  - **closed classes** -- prepositions, determiners, conjuctions, pronouns, particples
- POS Taggers try to find POS tags for the words.
- duck is a verb or noun? (morphological analyzer cannot make decision).
- A POS tagger may make that decision by looking the surrounding words.
  - Duck! (verb)
  - Duck is delicious for dinner. (noun)

Lexical Processing

- The purpose of lexical processing is to determine meanings of individual words.
- Basic methods is to lookup in a database of meanings – **lexicon**
- We should also identify non-words such as punctuation marks.
- Word-level ambiguity -- words may have several meanings, and the correct one cannot

be chosen based solely on the word itself.
  - bank in English

- Solution -- resolve the ambiguity on the spot by POS tagging (if possible) or pass-on the ambiguity to the other levels.

Syntactic Processing

- **Parsing** -- converting a flat input sentence into a hierarchical structure that corresponds to the units of meaning in the sentence.
- There are different parsing formalisms and algorithms.
- Most formalisms have two main components:
  - **grammar** -- a declarative representation describing the syntactic structure of sentences in the language.
  - **parser** -- an algorithm that analyzes the input and outputs its structural representation (its parse) consistent with the grammar specification.
- CFGs are in the center of many of the parsing mechanisms. But they are complemented by some additional features that make the formalism more suitable to handle natural languages.

Semantic Analysis

- Assigning meanings to the structures created by syntactic analysis.
- Mapping words and structures to particular domain objects in way consistent with our knowledge of the world.
- Semantic can play an import role in selecting among competing syntactic analyses and discarding illogical analyses.
  - I robbed the bank -- bank is a river bank or a financial institution
- We have to decide the formalisms which will be used in the meaning representation.

Knowledge Representation for NLP

- Which knowledge representation will be used depends on the application -- Machine Translation, Database Query System.
- Requires the choice of representational framework, as well as the specific meaning vocabulary (what are concepts and relationship between these concepts -- ontology)
- Must be computationally effective.

- Common representational formalisms: – first order predicate logic
  – conceptual dependency graphs – semantic networks
  – Frame-based representations

Discourse

- Discourses are collection of coherent sentences (not arbitrary set of sentences)
- Discourses have also hierarchical structures (similar to sentences)
- **anaphora resolution** -- to resolve referring expression – Mary bought a book for Kelly. **She** didn't like **it**.
  - **She** refers to Mary or Kelly. -- possibly Kelly
  - **It** refers to what -- book.
  – Mary had to lie for Kelly. **She** didn't like **it**.
- Discourse structure may depend on application. – Monologue
  – Dialogue
  – Human-Computer Interaction

Applications of Natural Language Processing

- Machine Translation – Translation between two natural languages. – See the Babel Fish translations system on Alta Vista.
- Information Retrieval – Web search (uni-lingual or multi-lingual).
- Query Answering/Dialogue – Natural language interface with a database system, or a dialogue system.
- Report Generation – Generation of reports such as weather reports.
- Some Small Applications –
  – Grammar Checking, Spell Checking, Spell Corrector

Machine Translation

- Machine Translation refers to converting a text in language A into the corresponding text in language B (or speech).
- Different Machine Translation architectures are: – interlingua based systems
  – transfer based systems
- Challenges are to acquire the required knowledge resources such as mapping rules and bi-lingual dictionary? By hand or acquire them automatically from corpora.
- Example Based Machine Translation acquires the required knowledge (some of it or all of it) from corpora.

FUZZY LOGIC SYSTEMS:

INTRODUCTION:

A large amount of data can constitute a proportionately large amount of information. But this comes with a level of uncertainty. As we come to know more, we also know how much we do not know and our awareness of the concept of complexity seems to increase. We tend to forego some precise data and allow uncertainty to creep into our perception. This is when we start describing things in a slightly vague and fuzzy manner.

Consider the case of the statement- "Drive Slowly". Does it mean you should drive at 10, 20 or 20.5 km/hour? The answer could be any value or a very different one depending on the context. If it is ascertained in a machine that any speed less than or equal to 20 km/hour means slow speed and anything above is fast, then does it mean that 20.1 km/hour is fast? This is an exaggeration in the real world. Fuzzy logic deals with how we can capture this essence of comprehension and embed it on the system by allowing for a gradual transition from slow to high speeds.

CRISP SETS:

The conventional machine uses crisp sets to take care of concepts like fast and slow speeds. It relates speed to crisp values thereby forming members that either belong to a group or do not belong to it.

For example:

Slow = {0, 5, 10, 15, 20, 25, 30, 35, 40}

We could mean a crisp set that says when the value of speed is equal to either of those mentioned in the set then the speed is categorized as slow. This may be modified to a closed interval [0, 40] to include the complete range of values. However, when the speed crosses over to 40.1 it will be categorized as not slow. Likewise 39.99 will be slow.

FUZZY SETS:

Fuzzy sets introduce a certain amount of vagueness to reduce complexity of comprehension. This set consists of elements that signify the degree or grade of membership to a fuzzy aspect. Membership values usually use closed intervals and denote the sense of belonging of a member of crisp set to a fuzzy set. To make the point clear consider a crisp set A comprising of elements that signify the ages of set of people in years.

A= {2, 4, 10.15,21, 30, 35, 40, 45, 60, 70}

We could classify age in terms of what are known as fuzzy linguistic variables- infant, child, adolescent, adult, young, and old.

A person whose age is 15 is no doubt young but how would you categorize a person who is 30.

If the latter is to be considered young what about the person who is 40? Is he old?

How do we translate all these into numbers for efficiently making the computer understand what our feelings about age are?

Inspect the below table giving ages and their membership to a particular set.

Table5.1: ages and their memberships

Age	Infant	Child	Adolescent	Young	Adult	Old
2	1	0	0	1	0	0
4	0.1	0.5	0	1	0	0
10	0	1	0.3	1	0	0
15	0	0.8	1	1	0	0
21	0	0	0.1	1	0.8	0.1
30	0	0	0	0.6	1	0.3
35	0	0	0	0.5	1	0.35
40	0	0	0	0.4	1	0.4
45	0	0	0	0.2	1	0.6
60	0	0	0	0	1	0.8
70	0	0	0	0	1	1

The values in the table indicate membership to the fuzzy sets – infant, child, adolescent, young, adult and old.

Thus a child of age 4 belongs only 50% to the fuzzy set child while when he is 10 years he is a 100% member. Note that membership is different from probabilities. Memberships do not necessarily add up to 1. The entries in the table have been made after a manual evaluation of the different ages.

SOME FUZZY TERMINOLOGY:

We could define some some terms based on a Universal set U.

Universe of Discourse (U):

This is defined as the range of all possible values that comprise the input to the fuzzy system.

Fuzzy Set:

Any set that empowers its members to have different grades of membership (based on membership function) in an interval [0, 1] is a fuzzy set.

Membership Function:

The membership function μ_A which forms the basis of a fuzzy set is given by

$$\mu_{A:\,U} \longrightarrow [0, 1]$$

Where the closed interval is one that holds real numbers.

Support of a fuzzy set (S_f):

The support S of a fuzzy set f, in a universal crisp set U is that set which contains all elements of the set U that have a non-zero membership value in f. for instance, the support of the fuzzy set adult is from table5.1

$S_{adlut} = \{21, 30, 35, 40, 45, 60, 70\}$

Depiction of a fuzzy set

A fuzzy set f in a universal crisp set U, is written as

$f= \mu_{1/s1+} \mu_{2/S2+} \mu_{3/S3+\ldots\ldots\ldots\ldots+} \mu_{n/Sn}$

Where μ_i is the membership and s_i is the corresponding term in the support set of f i.e., S_f.

This is however only a representation and has no algebraic implication (the slash and + signs do not have any meaning).

Accordingly,

Old = 0.1/21 + 0.3/30 +0.4/40 + 0.6/45 + 0.8/60 + 1/70

Fuzzy Set Operations:

- **Union:** The membership function of the union of two fuzzy sets A and B is defined as the maximum of the two individual membership functions. It is equivalent to the Boolean OR operation.

  $\mu_{AUB=} \max (\mu_A, \mu_B)$

- **Intersection:** The membership function of the intersection of two fuzzy sets A and B is defined as the minimum of the two individual membership functions. It is equivalent to the Boolean AND operation.

  $\mu_{AUB=} \min (\mu_A, \mu_B)$

- **Complement:** The membership function of the complement of a fuzzy set A is defined as the negation of the specified membership function μ_{A^-}. this is equivalent to the Boolean NOT operation

  $\mu_{A^-} = \mu_{AUB} = (1- \mu_A)$

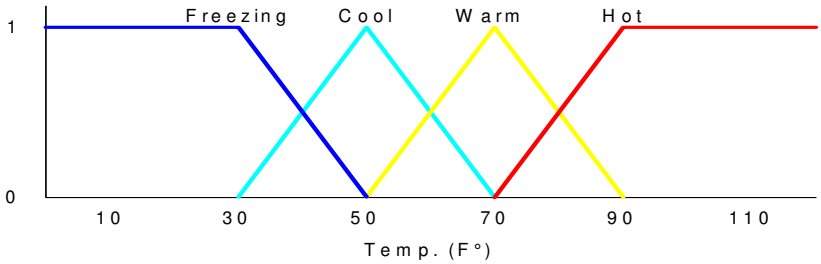
It may be further noted here that the laws of Associativity, Commutavity, Distributivity and De Morgon's laws hold in fuzzy set theory too.

Fuzzy Logic Control:

- Fuzzy Control combines the use of fuzzy linguistic variables with fuzzy logic
- Example: Speed Control
- How fast am I going to drive today?
- It depends on the weather.
- Disjunction of Conjunctions

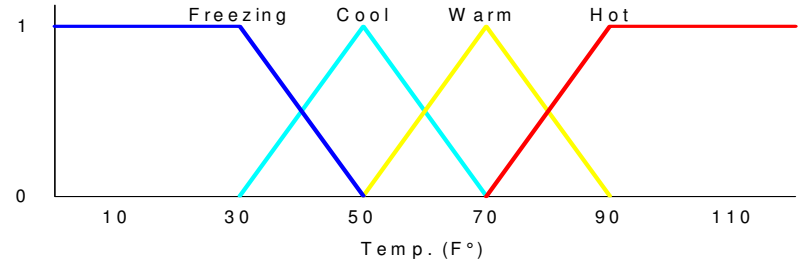
Inputs: Temperature

Temp: {Freezing, Cool, Warm, Hot}

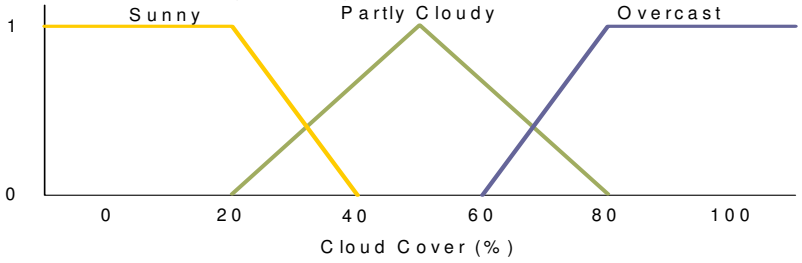


Inputs: Temperature, Cloud Cover

- Temp: {Freezing, Cool, Warm, Hot}

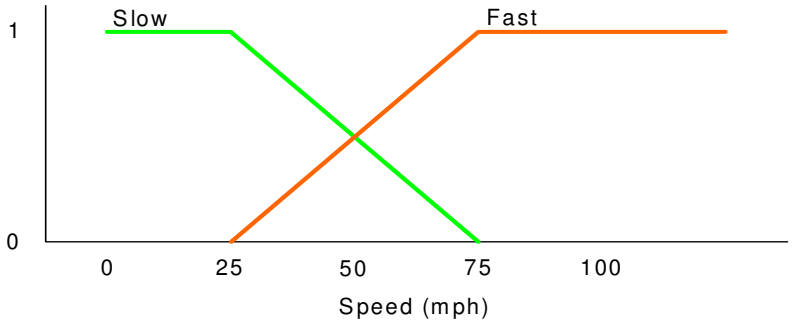


- Cover: {Sunny, Partly, Overcast}



Output: Speed

- Speed: {Slow, Fast}



Rules:

- If it's Sunny and Warm, drive Fast

Sunny(Cover)∧Warm(Temp)⇒ Fast(Speed)

- If it's Cloudy and Cool, drive Slow

Cloudy(Cover)∧Cool(Temp)⇒ Slow(Speed)

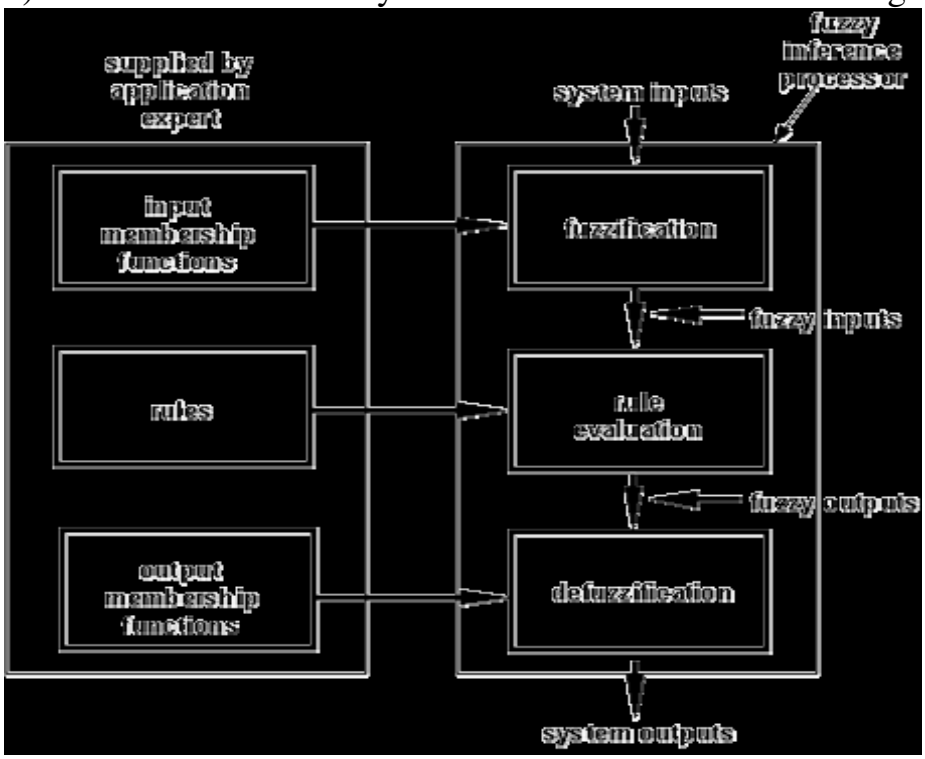
- Driving Speed is the combination of output of these rules...

Follow –up points:

- Fuzzy Logic Control allows for the smooth interpolation between variable centroids with relatively few rules.
- This does not work with crisp (traditional Boolean) logic.
- Provides a natural way to model some types of human expertise in a computer program.

Fuzzy Inferencing:

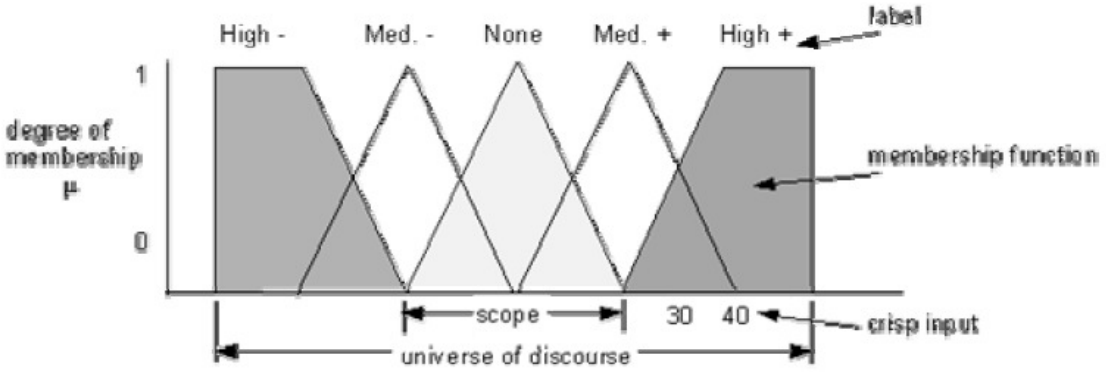
The process of fuzzy reasoning is incorporated into what is called a Fuzzy Inferencing System. It is comprised of three steps that process the system inputs to the appropriate system outputs. These steps are 1) Fuzzification, 2) Rule Evaluation, and
3) Defuzzification. The system is illustrated in the following figure.



Each step of fuzzy inferencing is described in the following sections.

Fuzzification

Fuzzification is the first step in the fuzzy inferencing process. This involves a domain transformation where crisp inputs are transformed into fuzzy inputs. Crisp inputs are exact inputs measured by sensors and passed into the control system for processing, such as temperature, pressure, rpm's, etc.. Each crisp input that is to be processed by the FIU has its own group of membership functions or sets to which they are transformed. This group of membership functions exists within a universe of discourse that holds all relevant values that the crisp input can possess. The following shows the structure of membership functions within a universe of discourse for a crisp input.



where:

degree of membership: degree to which a crisp value is compatible to a membership function, value from 0 to 1, also known as truth value or fuzzy input.

membership function, MF: defines a fuzzy set by mapping crisp values from its domain to the sets associated degree of membership.

crisp inputs: distinct or exact inputs to a certain system variable, usually measured parameters external from the control system, e.g. 6 Volts.

label: descriptive name used to identify a membership function.

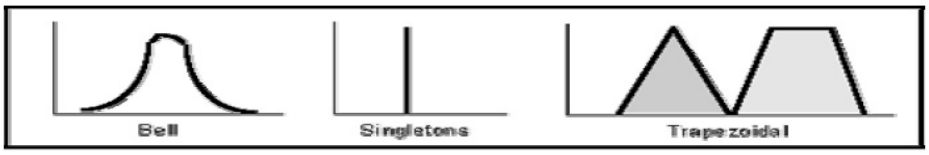
scope: or domain, the width of the membership function, the range of concepts, usually numbers, over which a membership function is mapped.

universe of discourse: range of all possible values, or concepts, applicable to a system variable.

When designing the number of membership functions for an input variable, labels must initially be determined for the membership functions. The number of labels correspond to the number of regions that the universe should be divided, such that each label describes a region of behavior. A scope must be assigned to each

membership function that numerically identifies the range of input values that correspond to a label.

The shape of the membership function should be representative of the variable. However this shape is also restricted by the computing resources available. Complicated shapes require more complex descriptive equations or large lookup tables. The next figure shows examples of possible shapes for membership functions.



Membership Function Shapes

When considering the number of membership functions to exist within the universe of discourse, one must consider that:

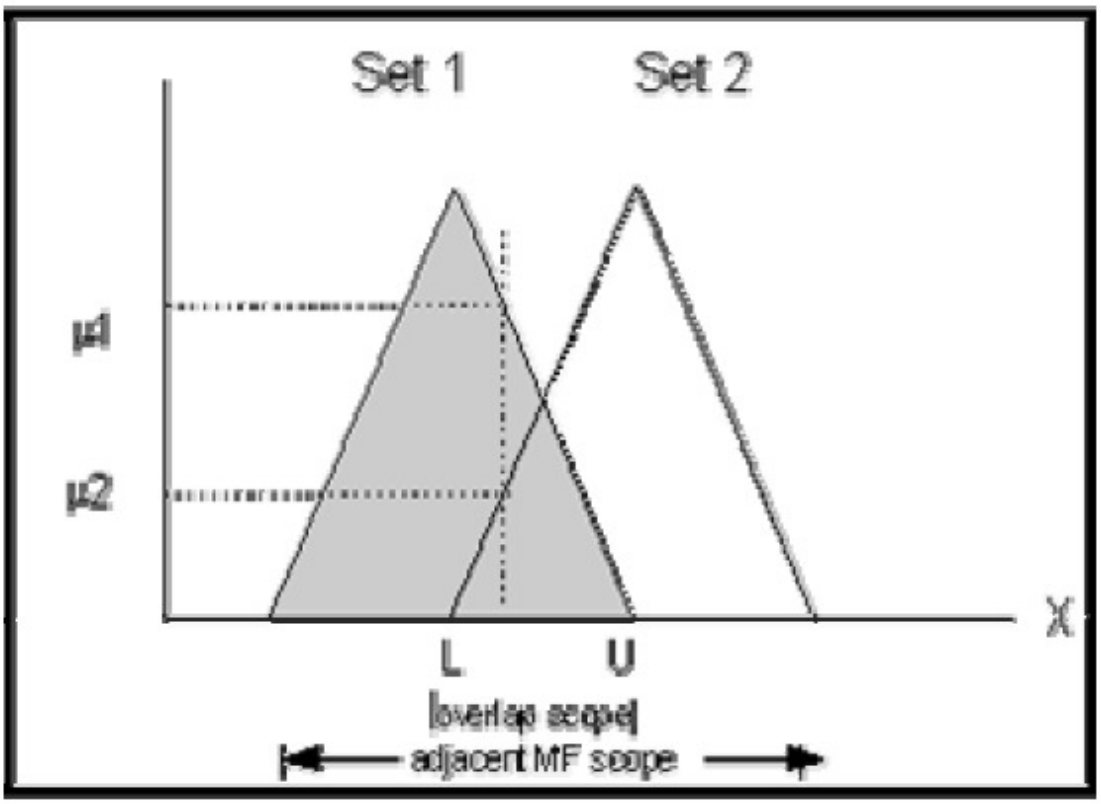
i) too few membership functions for a given application will cause the response of the system to be too slow and fail to provide sufficient output control in time to recover from a small input change. This may also cause oscillation in the system.

ii) too many membership functions may cause rapid firing of different rule consequents for small changes in input, resulting in large output changes, which may cause instability in the system.

These membership functions should also be overlapped. No overlap reduces a system based on Boolean logic. Every input point on the universe of discourse should belong to the scope of at least one but no more than two membership functions. No two membership functions should have the same point of maximum truth, (1). When two membership functions overlap, the sum of truths or grades for any point within the overlap should be less than or equal to 1. Overlap should not cross the point of maximal truth of either membership function. Marsh has proposed two indices to describe the overlap of membership functions quantitatively. These are overlap ratio and overlap robustness. The next figure illustrates their meaning.

$$\text{Overlap Ratio} = \frac{\text{overlap scope}}{\text{adjacent MF scope}} \qquad (1)$$

$$\text{Overlap Robustness} = \frac{\text{area of summed overlap}}{\text{max. area of summed overlap}} = \frac{\int_L^U (\mu_1 + \mu_2)dx}{2(U-L)} \qquad (2)$$



Overlap Indices

The fuzzification process maps each crisp input on the universe of discourse, and its intersection with each membership function is transposed onto the μ axis as illustrated in the previous figure. These μ values are the degrees of truth for each crisp input and are associated with each label as fuzzy inputs.

Defuzzification

Defuzzification involves the process of transposing the fuzzy outputs to crisp outputs. There are a variety of methods to achieve this, however this discussion is limited to the process used in this thesis design.

A method of averaging is utilized here, and is known as the Center of Gravity method or COG, it is a method of calculating centroids of sets. The output membership functions to which the fuzzy outputs are transposed are restricted to being singletons. This is so to limit the degree of calculation intensity in the microcontroller. The fuzzy outputs are transposed to their membership functions similarly as in fuzzification. With COG the singleton values of outputs are calculated using a weighted average, illustrated in the next figure. The crisp output is the result and is passed out of the fuzzy inferencing system for processing elsewhere.

Fuzzy HEDGES:

Another important feature of fuzzy systems is the ability to define "hedges," or modifier of fuzzy values. These operations are provided in an effort to maintain close ties to natural language, and to allow for the generation of fuzzy statements through mathematical calculations. As such, the initial definition of hedges and operations upon them will be quite a subjective process and may vary from one project to another. Nonetheless, the system ultimately derived operates with the same formality as classic logic.

The simplest example is in which one transforms the statement "Jane is old" to "Jane is very old." The hedge "very" is usually defined as follows:

m"very"A(x) = mA(x)^2

Thus, if mOLD(Jane) = 0.8, then mVERYOLD(Jane) = 0.64.

Other common hedges are "more or less" [typically SQRT(mA(x))], "somewhat," "rather," "sort of," and so on. Again, their definition is entirely subjective, but their operation is consistent: they serve to transform membership/truth values in a systematic manner according to standard mathematical functions.

A more involved approach to hedges is best shown through the work of Wenstop in his attempt to model organizational behavior. For his study, he constructed arrays of values for various terms, either as vectors or matrices. Each term and hedge was represented as a 7-element vector or 7x7 matrix. He ten intuitively assigned each element of every vector and matrix a value between 0.0 and 1.0, inclusive, in what he hoped was intuitively a consistent manner. For example, the term "high" was assigned the vector

0.0 0.0 0.1 0.3 0.7 1.0 1.0

and "low" was set equal to the reverse of "high," or

1.0 1.0 0.7 0.3 0.1 0.0 0.0

Wenstop was then able to combine groupings of fuzzy statements to create new fuzzy statements, using the APL function of Max-Min matrix multiplication. These values were then translated back into natural language statements, so as to allow fuzzy statements as both input to and output from his simulator. For example, when the program was asked to generate a label "lower than sortof low," it returned "very low;" "(slightly higher) than low" yielded "rather low," etc. The point of this example is to note that algorithmic procedures can be devised which translate "fuzzy" terminology into numeric values, perform reliable operations upon those values, and then return natural language statements in a reliable manner.