Uncertainty

Problems often have a certain amount of uncertainty, possibly due to:

- Incompleteness of information about the environment,
  - E.g., loss of sensory information such as vision
- Incorrectness in the interpretation of the environment
  - Information can be random, vague, imprecise

Dealing with uncertainty is an important aspect of intelligence!

Expert Systems for Uncertainty

- Possible to deal with uncertainty by using "expert systems."

- An expert system is a program with highly domain specific knowledge which provides "expert quality" performance in a specific problem domain.
  - Usually focused on a narrow set of problems (**specialization**).
  - Knowledge can be **both theoretical** and **practical**; information is obtained from human experts which have developed rules of thumb, heuristics, etc. in addition to theoretical knowledge.

- Expert Systems are **practical programs** which use heuristics developed by humans to solve specific problems.
  - They can use imperfect knowledge to obtain useful solutions to problems.

Capabilities and motivation

- Expert Systems generally support things like:
  - Inspection of the reasoning process;
    - If we (as humans) are to trust an expert system, we must be able to query and examine the path to conclusions (and be satisfied with it).
  - Addition and/or deletion of skills in the knowledge base;
    - We must be able to refine and debug the expert system.
- Expert Systems are nice in the sense that they allow for the distribution of expert knowledge in the form of a computer program.
  - Experts in any given area are costly and scarce.
  - Retraining of people is hard and/or costly.
  - Some problems have too much information and require fast timely solutions.

Typical uses for expert systems

Typical uses:

- **Planning**: deriving a sequence of actions to achieve goals given initial conditions and runtime constraints.
- **Diagnosis**: cause of malfunctions based on observable symptoms.
- **Debugging and repair**: prescribing and implementing remedies for malfunctions.
- **Instruction**: detecting and correcting deficiencies in learning.
- **Control**: governing the behavior of a complex system.
- **Design**: selection of components to meet performance goals and constraints.

Measuring performance of an expert system

- The performance of an expert system is typically measured in terms of something like a Turing Test,
  - Blind comparison between system and expert.

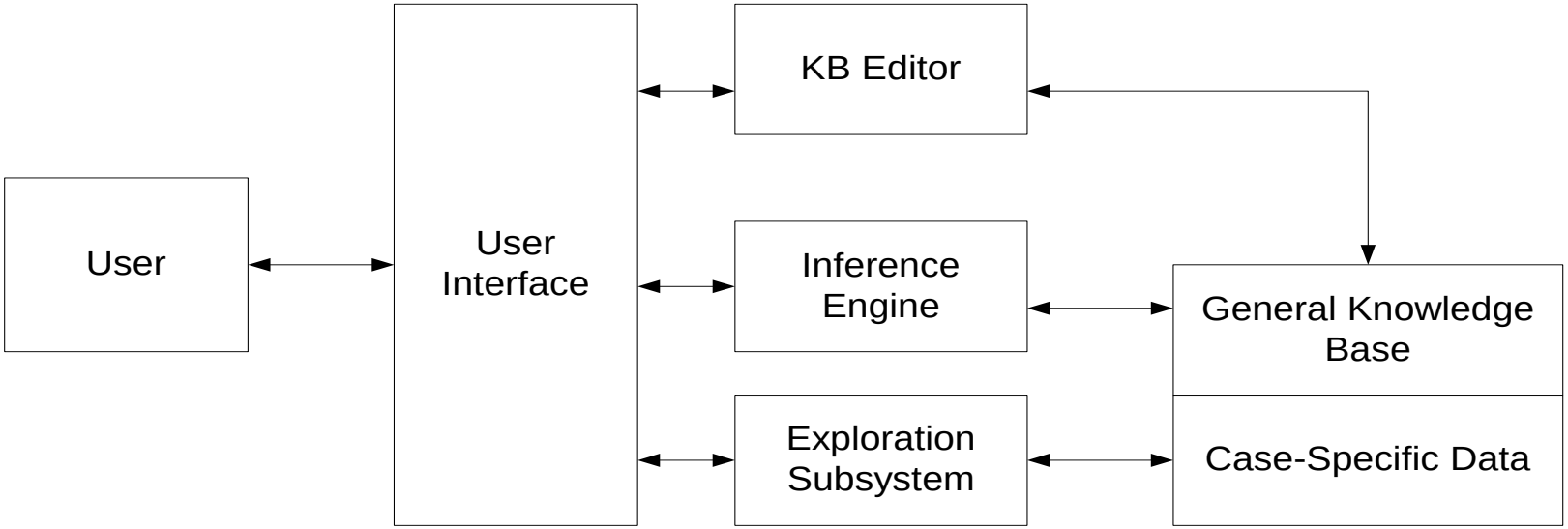
- E.g., **MYCIN** is an expert system for medical diagnosis of infectious blood diseases.

- Ask MYCIN and real doctors blindly about numerous symptoms and prescribed treatment.

  - Don't know if the answer is coming from the Expert System or a doctor

- Does MYCIN perform as well as doctors? The answer was **yes**.

Generic architecture of an expert system



Components of an expert system

- **General Knowledge Base**:
  - Contains problem solving knowledge for a particular problem domain.
  - Often stated as **"if…then.."** rules (**a rule-based expert system**).
- **Case Specific Data**:
  - Facts, conclusions and information relevant to the specific problem instance under consideration.
  - Changes from problem to problem.
  - Knowledge obtained by **querying** user or by **inferencing** on other case specific information.
- **Knowledge Base Editor**:
  - Used to locate and correct bugs; add updates to the knowledge base.

Components of an expert system

- **User Interface**:
  - Simplifies computer-human interactions; hides system complexity, translation of results into English, etc.
- **Inference Engine**:
  - Mechanism for applying knowledge to the solution of an actual problem.
  - Can be viewed as an interpreter for the knowledge base.
  - Derives new information from the general knowledge base and the case-specific information.
- **Exploration Subsystem**:
  - Allows the user to ask the Expert System to explain things;
  - e.g., reason or justification for a conclusion, explanation of why data is requested, etc.

Expert system shells

- Common to bundle together the user interface, the knowledge base editor, the inference engine and the exploration subsystem.
  - Idea is to separate common (and generic) features of an expert system;
  - Allows for more rapid prototyping of new expert systems.

When Are Expert Systems Suitable?

- **Needs justify costs.**
  - e.g., PROSPECTOR was an expert system used for locating mineral deposits without the expensive drilling tests, etc.
- **Human expertise is not available where needed**
  - e.g., Medical diagnosis (this is hard to keep up to date)
- **Problem solvable using symbolic reasoning**
  - e.g., Don't need manual or perceptual skill.

When Are Expert Systems Suitable?

- **Problem not solvable using traditional mathematical methods**
  - An expert system will not likely beat solid mathematics.
- **Cooperative and articulate experts exist**
  - Can't build an expert system without experts willing to help, able to convey their knowledge.
- **Problem is of proper size and scope**
  - e.g., can't replace a doctor, but can help in particular cases
  - problem might take a long time due to computations involved.

Fuzzy Logic

- Initiated in 1965 by Dr. L. A. Zadeh.
- A multi-valued logic that allows intermediate values to be defined;
  - e.g., rather than simply using TRUE/FALSE, notions like "tall" or "fast" can be formulated mathematically and processed via computer.
- Fuzzy logic and fuzzy systems are one way to deal with uncertainty.
  - Provides a simple way to represent and reason with vague, ambiguous, imprecise and noisy input or knowledge.
  - Incorporates a rule-based approach* to solving a control problem rather than modeling the problem mathematically.

Linguistic

- Fuzzy logic:
  - A way to represent variation or imprecision in logic
  - A way to make use of natural language in logic
  - Approximate reasoning
- Humans say things like "If it is sunny and warm today, I will drive fast"
- Linguistic variables:
  - Temp: {freezing, cool, warm, hot}
  - Cloud Cover: {overcast, partly cloudy, sunny}
  - Speed: {slow, fast}

Crisp (Traditional) Variables

- Crisp variables represent precise quantities:
  - x = 3.1415296
  - $A \in \{0,1\}$
- A proposition is either True or False
  - $\boldsymbol{A \wedge B \Rightarrow C}$
- King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
- Richard is either greedy or he isn't:
  - Greedy(Richard) $\in \{0,1\}$

Distinction from probability

- Important to note that fuzzy logic and probability are different.
- Both operate over the same numerical range and have similar values; e.g., 0.0 represents false (non-membership) and 1.0 represents true (full-membership).
- However, probability says things like "there is a 50% chance that x is y" while fuzzy terminology corresponds to x's degree of membership in y.
- Semantic difference, and is significant:
  - probabilistic view: x is or is not in y; we only have a 50% chance of knowing which.
  - By contrast, fuzzy terminology supposes that x is "more or less" in y.

Fuzzy Sets

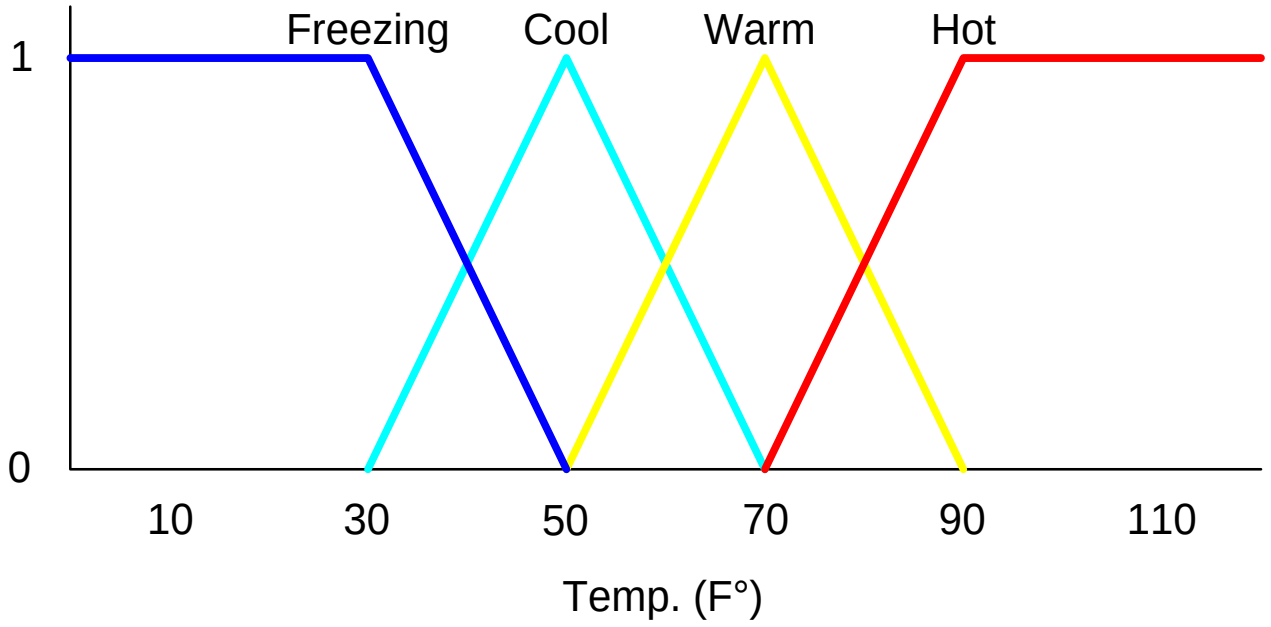
- What if Richard is only somewhat greedy?
- Fuzzy Sets can represent the degree to which a quality is possessed.
- Fuzzy Sets (Simple Fuzzy Variables) have values in the range of [0,1]
- Greedy(Richard) = 0.7

Fuzzy Linguistic Variables

- Fuzzy Linguistic Variables are used to represent qualities spanning a particular spectrum
- Temp: {**Freezing**, **Cool**, **Warm**, **Hot**}
- Membership Function
- Question: What is the temperature?
- Answer: It is warm.
- Question: How warm is it?

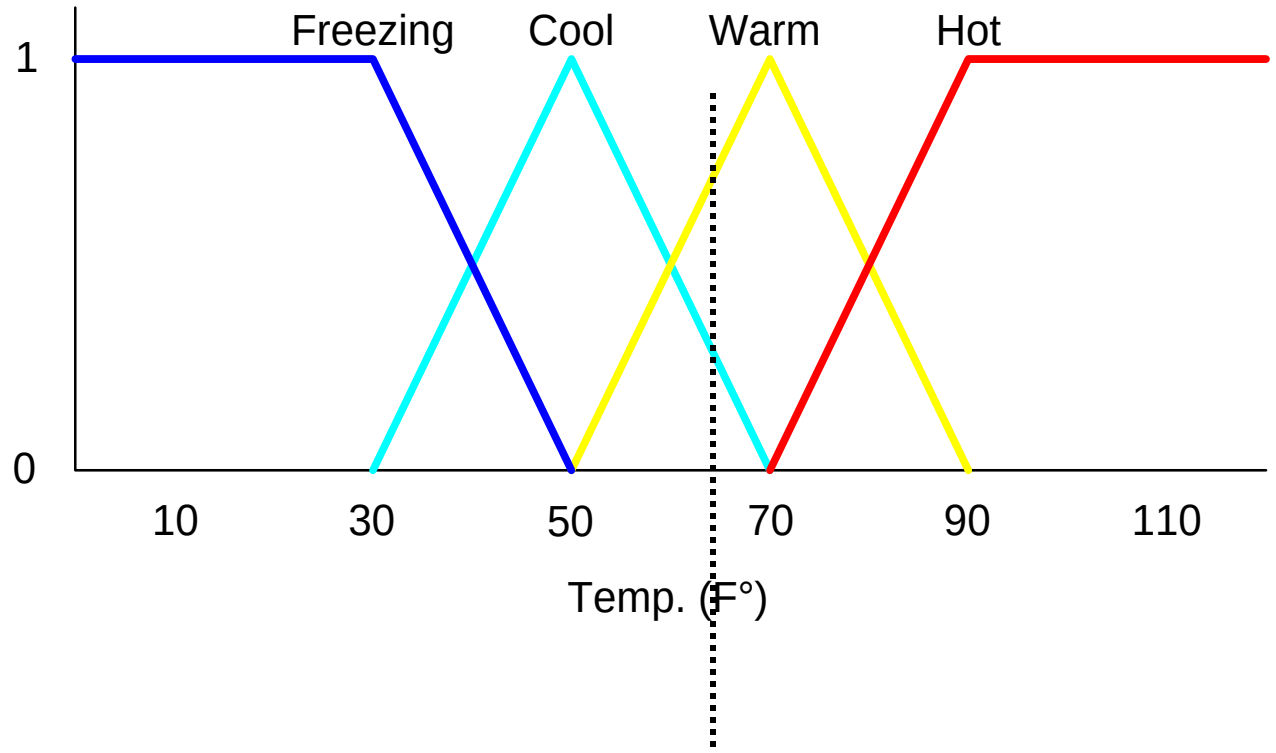
Membership Functions

- Temp: {Freezing, Cool, Warm, Hot}
- Degree of Truth or "Membership"



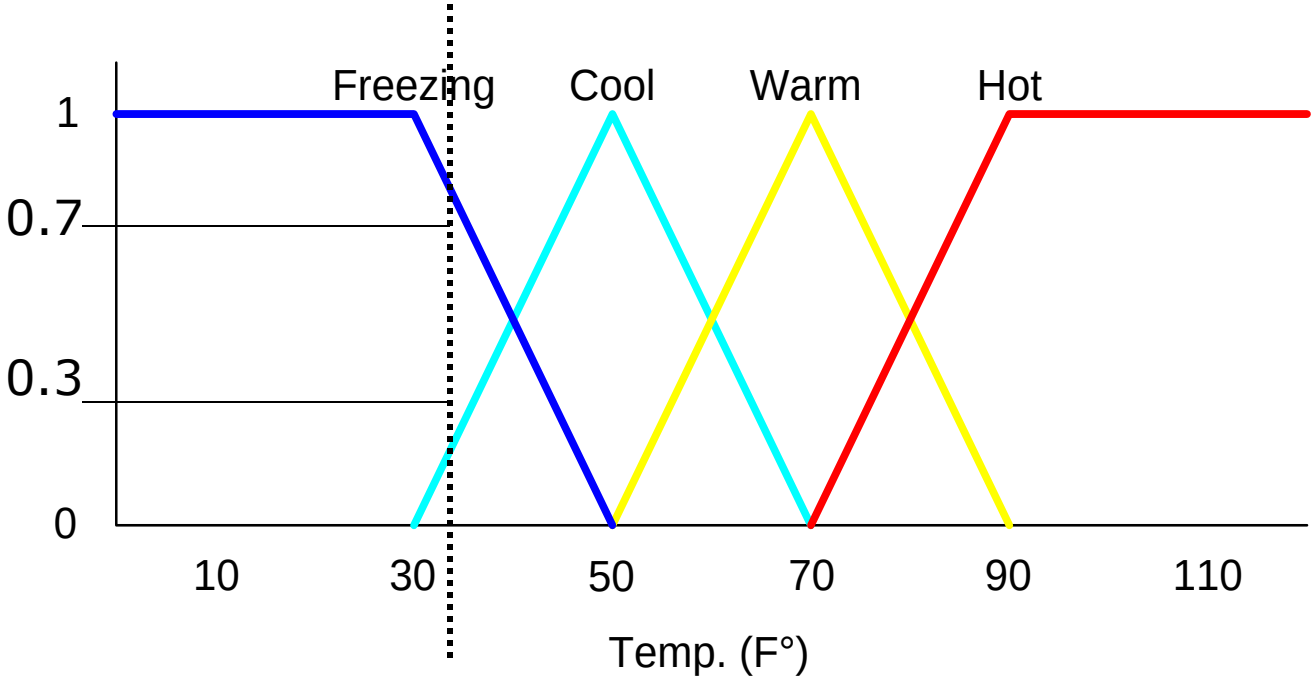
Membership Functions

- How cool is 36 F° ?



Membership Functions

- How cool is 36 F° ?
- It is 30% Cool and 70% Freezing



Operations on fuzzy sets

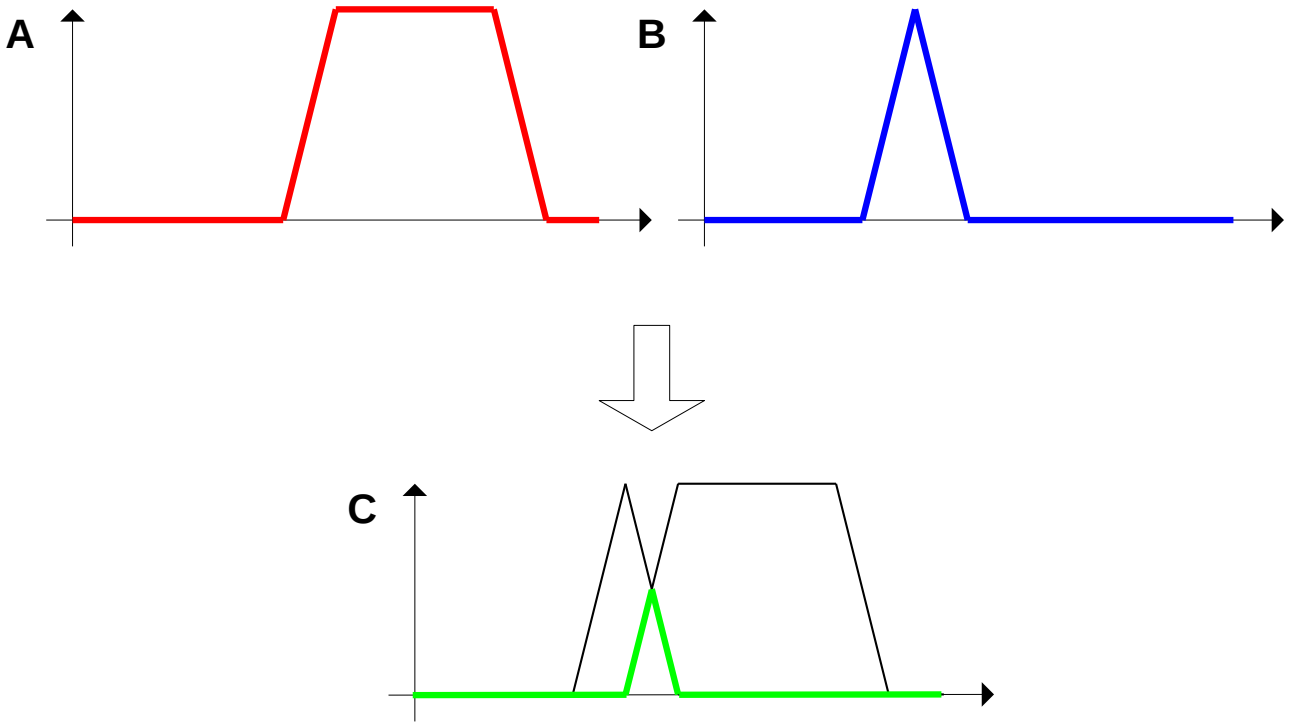
- Can introduce basic operations on fuzzy sets similar to those operations for crisp sets.

- Consider three simple (basic) set operations: intersection, union and negation.

  - In particular, look at the AREA of the memberships functions …

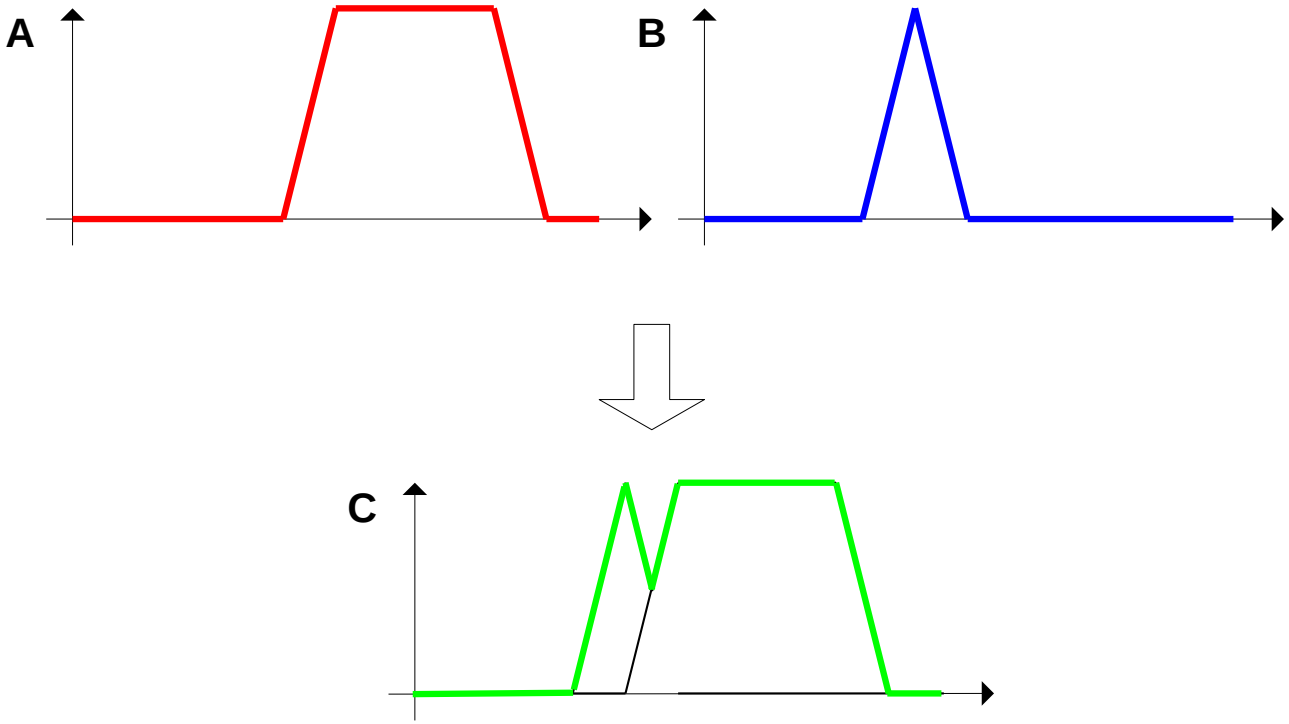
Intersection of fuzzy sets

- Zadeh suggested the MIN operation for intersection; e.g., given two fuzzy sets **A** and **B**, then **C = A \ B** with $\mu_C(x) = \mathbf{MIN}\{\mu_A(x), \mu_B(x)\}$, **x 2 U**.



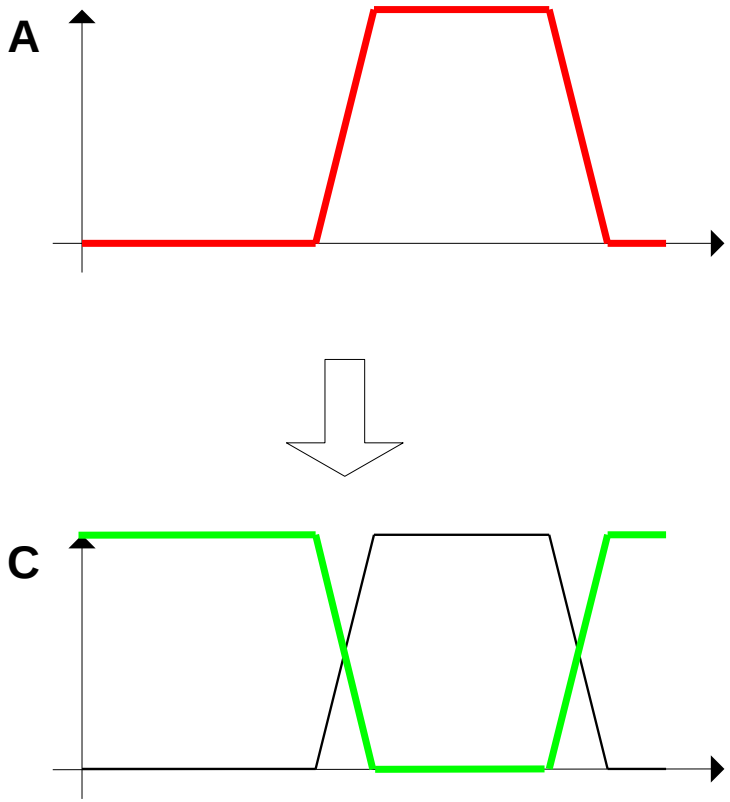
Union of fuzzy sets

- Zadeh suggested the MAX operation for union; e.g., given two fuzzy sets ***A*** and **B**,then ***C*** = ***A*** **[** **B** with $\mu_C(x) = MAX\{\mu_A(x), \mu_B(x)\}$, **x 2 U.**



Negation/Complement of fuzzy sets

- Given a fuzzy set ***A***, then ***C*** = : ***A*** is the complement of ***A*** with $\mu_{:A}(x) = 1.0 - \mu_A(x),\ x \in U$

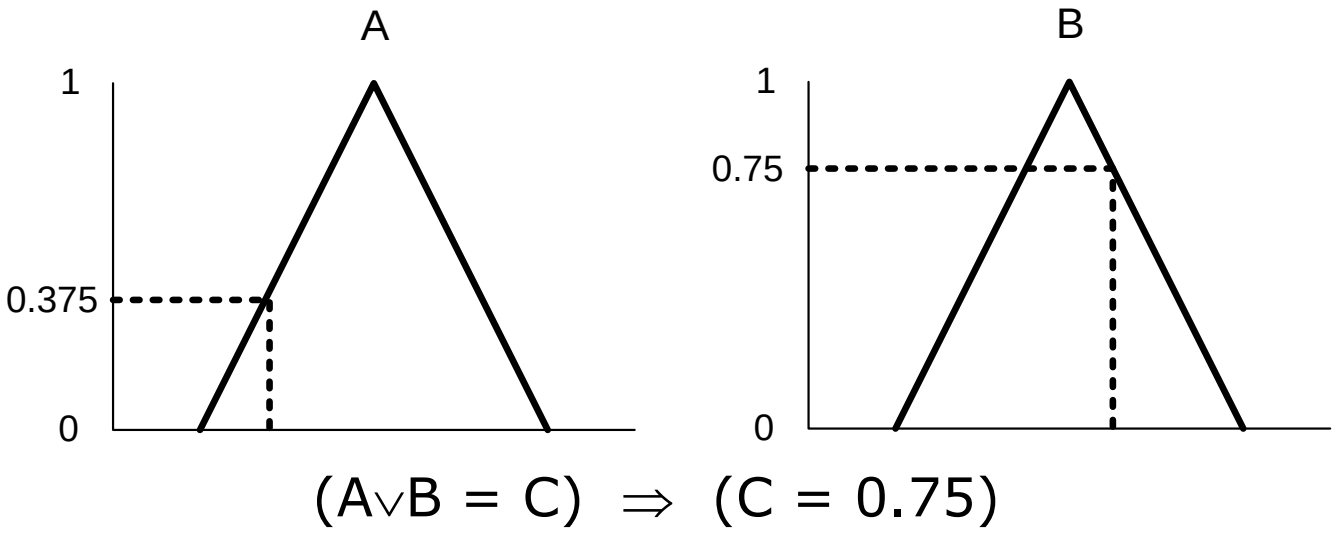


Fuzzy Logic

- How do we use fuzzy membership functions in predicate logic?
- Fuzzy logic Connectives:
  - Fuzzy Conjunction, $\wedge$
  - Fuzzy Disjunction, $\vee$
- Operate on degrees of membership in fuzzy sets

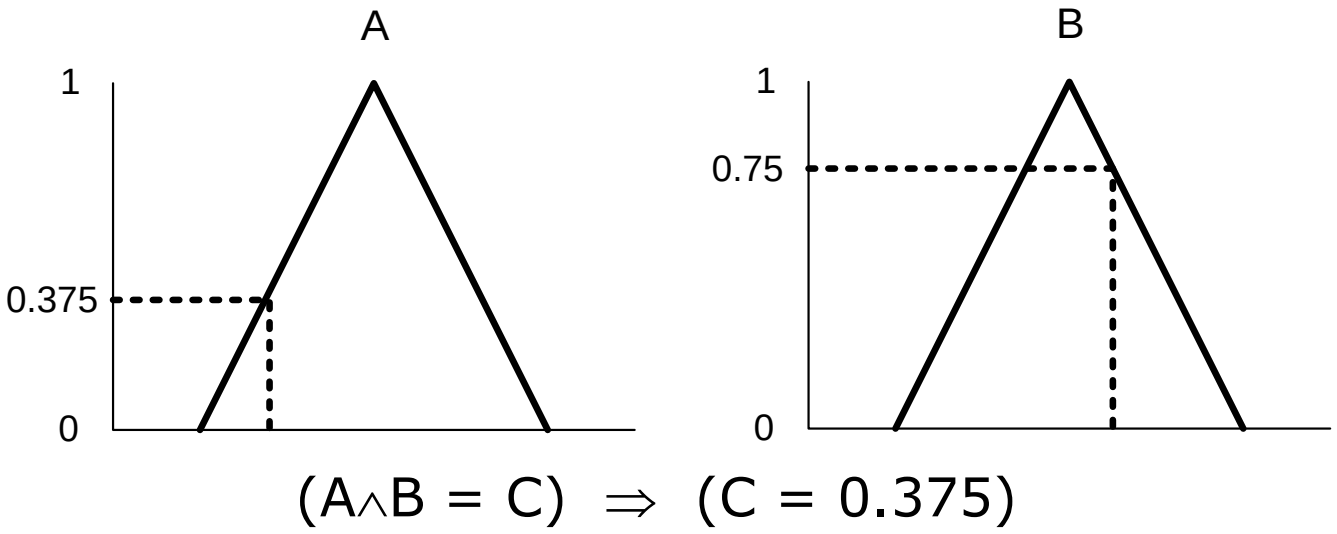
Fuzzy Disjunction

- $A \vee B \triangleq \max(A, B)$
- $A \vee B = C$ "Quality C is the disjunction of Quality A and B"



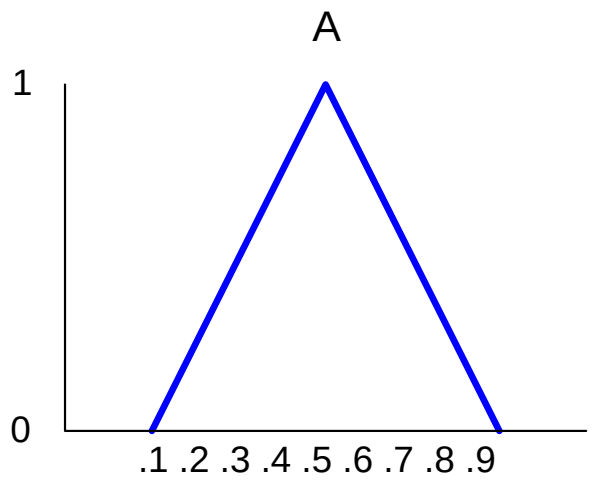
Fuzzy Conjunction

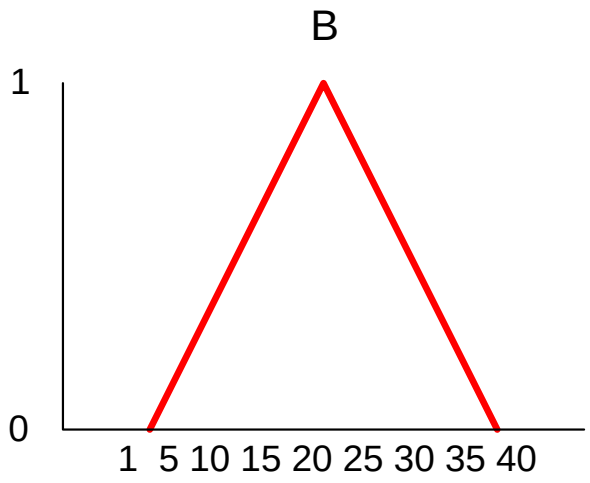
- $A \wedge B \triangleq \min(A, B)$
- $A \wedge B = C$ "Quality C is the conjunction of Quality A and B"



Example: Fuzzy Conjunction

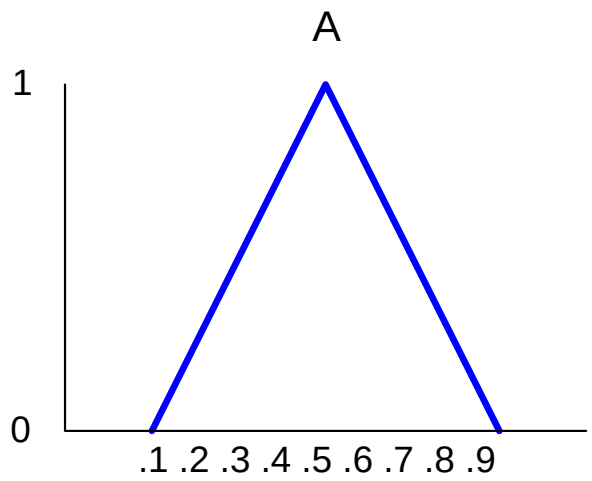
Calculate A∧B given that A is .4 and B is 20

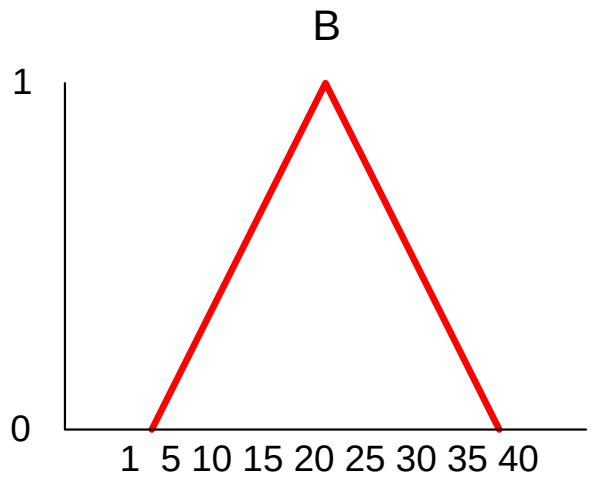




Example: Fuzzy Conjunction

Calculate A∧B given that A is .4 and B is 20

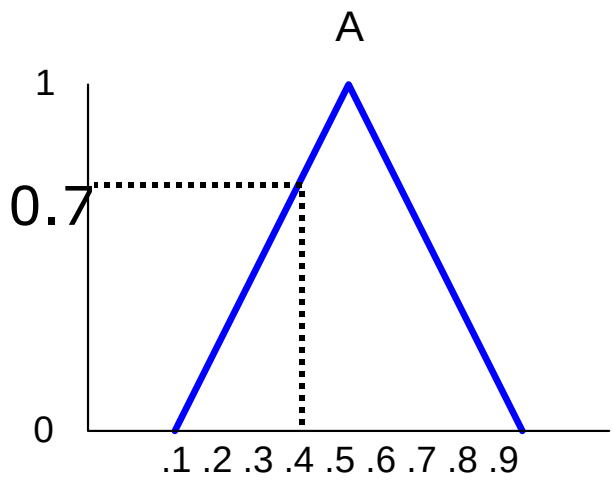


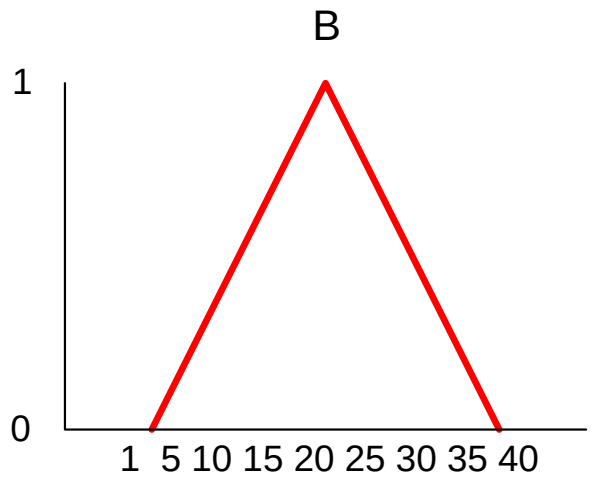


Determine degrees of membership:

Example: Fuzzy Conjunction

Calculate A∧B given that A is .4 and B is 20



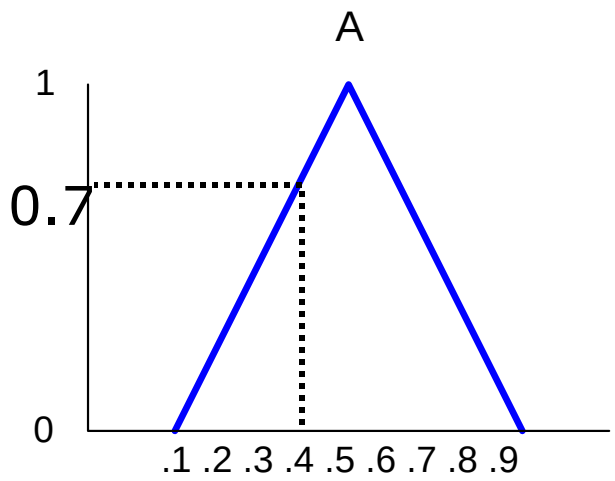


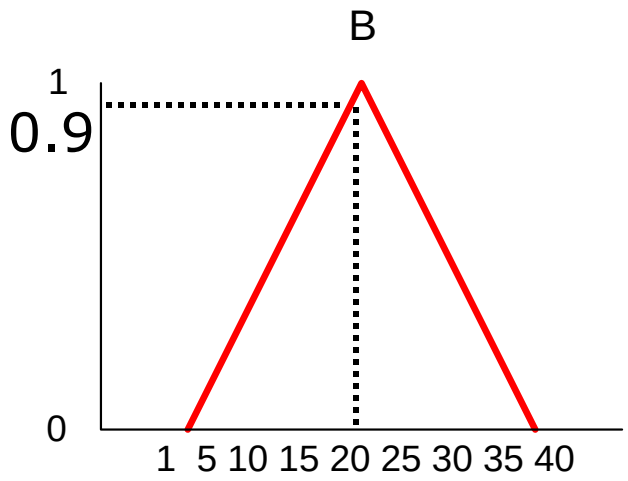
Determine degrees of membership:

A = 0.7

Example: Fuzzy Conjunction

Calculate A∧B given that A is .4 and B is 20



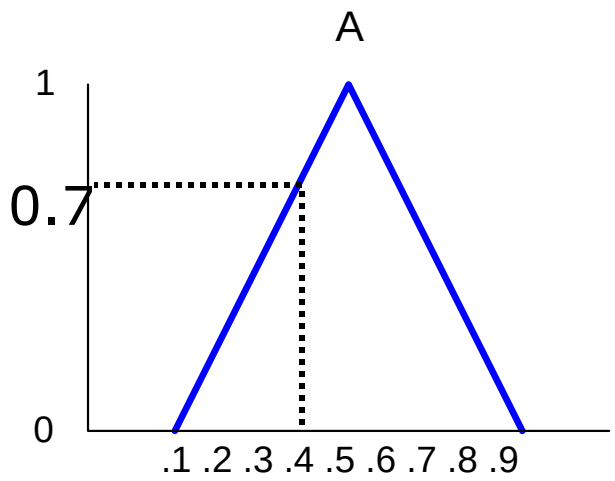


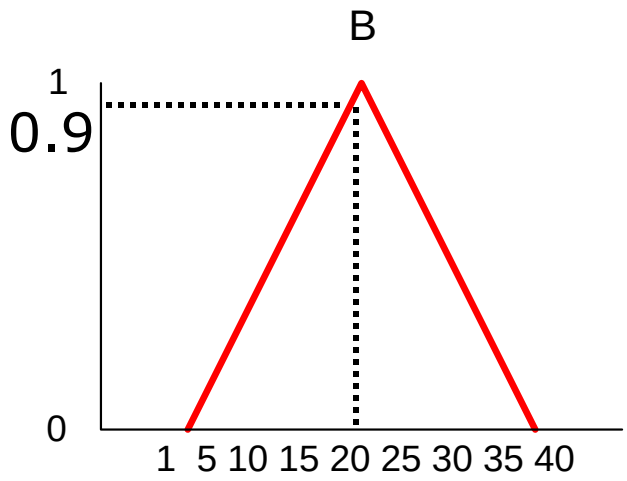
Determine degrees of membership:

A = 0.7 B = 0.9

Example: Fuzzy Conjunction

Calculate A∧B given that A is .4 and B is 20





Determine degrees of membership:

A = 0.7 B = 0.9

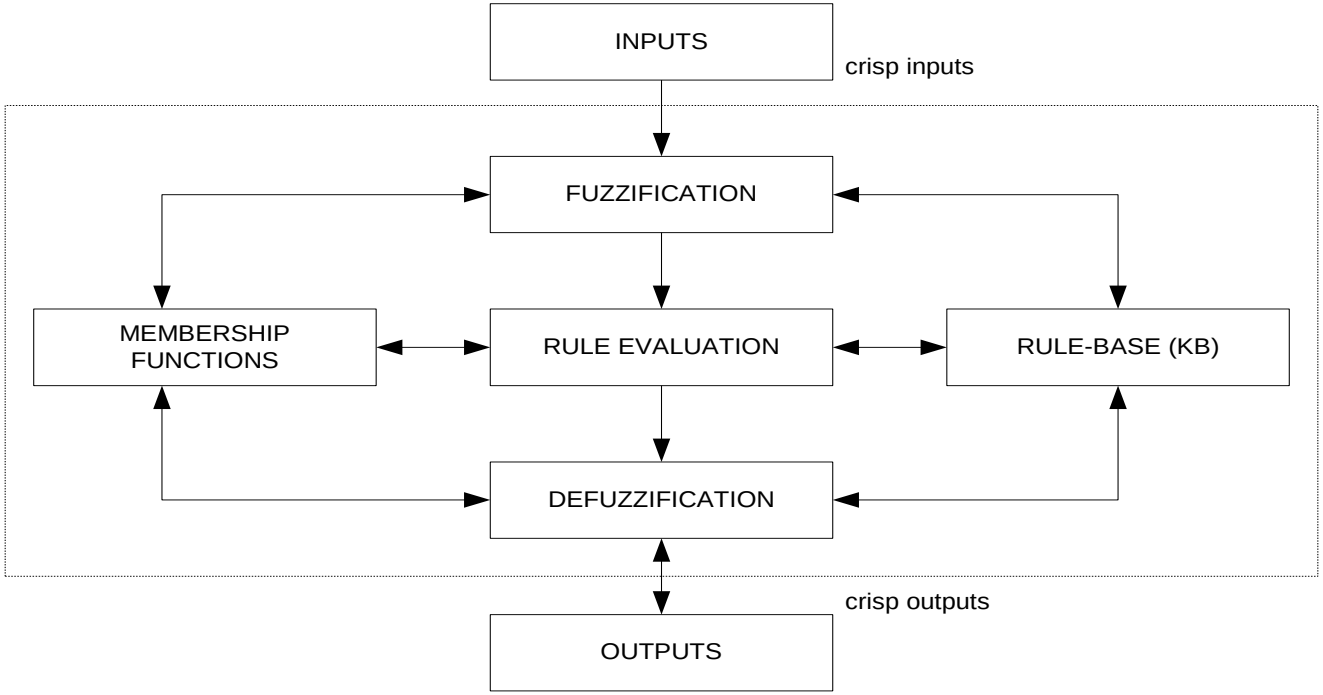
Apply Fuzzy AND

A∧B = min(A, B) = 0.7

Fuzzy expert system

- Can use fuzzy sets and fuzzy set properties to build **fuzzy rule-based systems**; popular for controlling the behavior of a complex system.
  - Given input(s) and a set of rules (describing how to react), what should the output(s) of the system be?
  - The fuzziness allows for the use of rules of thumb to describe "if-then" type rules.

Topology of a fuzzy rule-based system



- Inputs and outputs are CRISP values since they represent a measurement, setting, decision, etc.
- Internals of the system work using fuzzy membership functions to handle uncertainty.

Fuzzification

- Inputs to a fuzzy system are going to be crisp values; e.g., Crisp measurements taken from the system/environment such as position, velocity, temperature, etc.

- Each input will have an associated domain of possible values and degrees of membership in one or more fuzzy sets.

- **Fuzzification** is the process of computing the degree of membership of each current input value in the appropriate fuzzy sets.
  - So, fuzzification of input values is simple; we simply compute the degree of memberships in the appropriate fuzzy sets.

- **The fuzzification is needed since inferencing (rule evaluation) requires degrees of membership in the premises of the rules.**

Defuzzification

- After we apply our rules to the fuzzified input values, we will find that each output has a degree of membership in a number of fuzzy sets.

- However, each output must be a crisp value; e.g., force = 200 newtons.

- **Defuzzification** is the process of computing a **crisp value** for an output **from** the (possibly) many **degrees of membership values of an output** in different fuzzy sets.

Defuzzification

Defuzzification is needed for two main reasons:

- To **decipher the meaning of vague expressions** like "force is positive small", "force is negative large", "order is sell", "order is hold", etc. **thereby obtaining a crisp output value** that is useable.

- To **resolve conflicts** between competing actions that result **from the evaluation of different rules**.

  - Conflicting rules, for instance, might be something like: one rules says "force is positive small" and another rule says "force is positive large".

Defuzzification using weighted centroids

- There are several methods for computing a crisp output value given degrees of memberships in several fuzzy sets.

- One method is the so called "center of gravity approach" (centroid method):

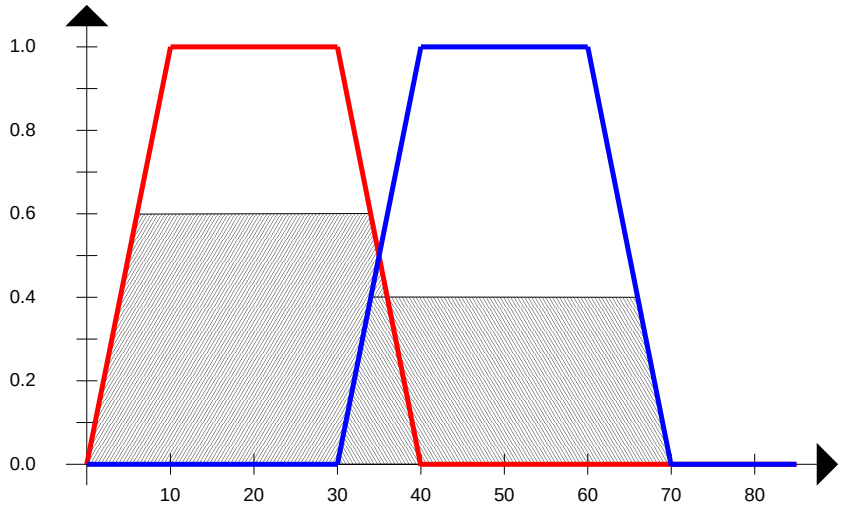
  1. Compute centroids for output membership functions.

  2. Limit height of each membership function by the degree of membership of the output and compute the resulting area.

  3. Compute the weighted average of the centroids using computed areas.

Defuzzification using weighted centroids

- Degree of membership is 0.6 and 0.4.
- Centroid of membership functions is 20 and 50.
- Areas are 0.6(40+28)/2 = 20.4 and 0.4(40+32)/2 = 14.4
- Crisp output is (20.4(20) + 14.4(50)) / (20.4+14.4) = 32.4



Fuzzy rules and rule-bases

- Generally expressed in the form:

  IF (PREMISE)
  THEN (CONCLUSION)

- Input values (fuzzified) and fuzzy sets/membership values for inputs are used in the premise.

- Fuzzy sets/membership values for outputs are used in the conclusion.

- Example from some sort of control problem in which the input comes from sensors measuring angular displacement and velocity; output is some sort of applied force:

  IF "angle is positive large" AND "velocity is negative small"
  THEN "force is positive small".

Fuzzy rules and rule-bases

- Dissection of rule:

  IF "angle is positive large" AND "velocity is negative small"
  THEN "force is positive small".

- Variables "angle" and "velocity" are crisp measured input values. Variable "force" is a crisp output value.

- Terms "positive large" and "negative small" are fuzzy sets for "angle" and "velocity", respectively. "positive small" is a fuzzy set for "force".

- Terms "angle is positive large" is the degree of membership of "angle" in "positive large"; "velocity is negative small" is the degree of membership of "velocity" in "negative small"; "force is positive small" is the degree of membership of "force" in the fuzzy set "positive small".

Inferencing and rule evaluation

- Clearly, every input and output will have multiple fuzzy sets associated with it. Similarly, every fuzzy system will have multiple rules.

- Need to perform **inferencing** given the set of **IF**… **THEN**… rules that describe the behavior of the system; i.e.,
  - **Given degrees of memberships of the premises, compute degrees of memberships of the conclusions.**

- Define the strength of any/every rule as the minimum of the premises (antecedents):*
  - **RULE_STRENGTH(RULE) = MIN(ANTECEDENT1,ANTECEDENT2, … , ANTECEDENTn)**

- Recall that the antecedents are degrees of membership of the inputs in fuzzy sets.

Strength of rules and output membership

- Consider the following rule for stock trading:

  IF "share price is dropping " AND "trading volume is light"
  THEN "action is hold".

- Assume "share price" fuzzifies to 0.2 in the fuzzy set "dropping" (i.e., "share price is dropping" = 0.2) and "trading volume" fuzzifies to 1.0 in the fuzzy set "light" (i.e., "trading volume is light" = 1.0);
  - The rule strength is MIN(0.2,1.0) = 0.2.

- **The degree of membership of the output is assigned the strength of the rule;**
  - So, the output "action" in the fuzzy set "hold" is 0.2 ("action is hold" = 0.2).

- Rule strength is the minimum to avoid making bad decisions in the sense that a rule is **weak** if **any** of its antecedents are **not very true**.

- **What if multiple rules have the same conclusion, but generate a different degree of membership???**

Multiple rules with identical conclusions

- Consider the two rules with the same conclusion:

  IF "share price is dropping " AND "trading volume is light"
  THEN "action is hold".

  IF "share price is stable" AND "trading volume is light"
  THEN "action is hold".

- If "share price is stable" = 0.9, "trading volume is light" = 1.0 and "share price is dropping" = 0.2:
  - Rule#1: has strength MIN(0.2,1.0) = 0.2;
  - Rule#2: has strength MIN(0.9,1.0) = 0.9.

- **So what value should "action is hold" be assigned?**
  - **i.e., what is the degree of membership of "action" in the fuzzy set "hold"?**

Max-min rule

- Since we used MIN for the premises (to determine rule strength) we will use MAX for multiple identical conclusions.

- We can think of taking the maximum of the rule strengths as taking the conclusion of the "most true rule".*
  - So in our previous example, "action is hold" = 0.9; i.e., "action" has degree of membership 0.9 in the fuzzy set "hold".

- Hence the name of the rule…
  - Since the degree of membership for an output is taken as the **maximum** of rule strengths, which themselves are the **minima** of the antecedents, choosing output membership this way is the **MAX-MIN RULE**.

- After applying inferencing (rule-evaluation) and the max-min rule, we will have a membership value for the output variables in each of their associated fuzzy sets; we can proceed to defuzzification.

Fuzzy Control

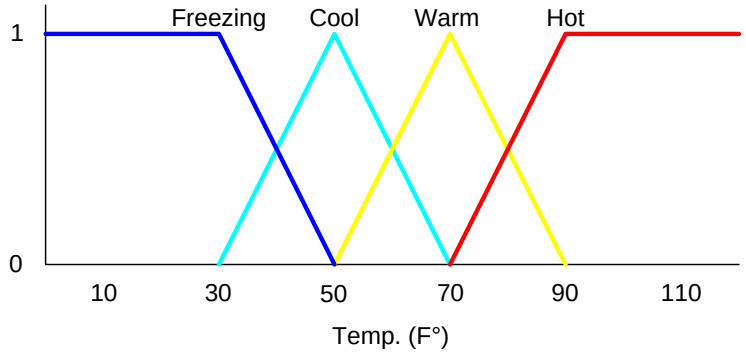
- Fuzzy Control combines the use of fuzzy linguistic variables with fuzzy logic
- Example: Speed Control
- How fast am I going to drive today?
- It depends on the weather.
- Disjunction of Conjunctions

- Crisp: When you are at 10 metres from the junction start braking at 50% pedal level.

- Fuzzy: When you are near the junction, start braking slowly.

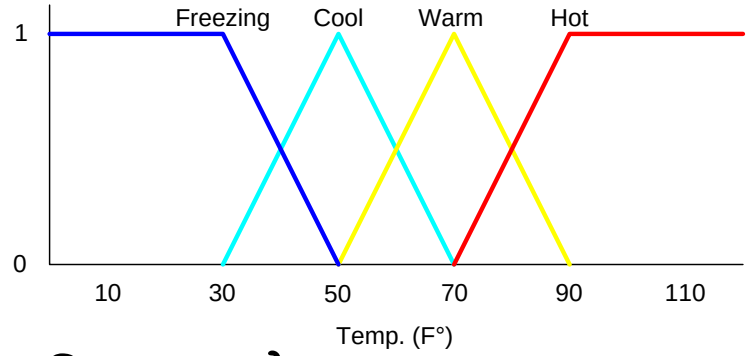
Inputs: Temperature

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

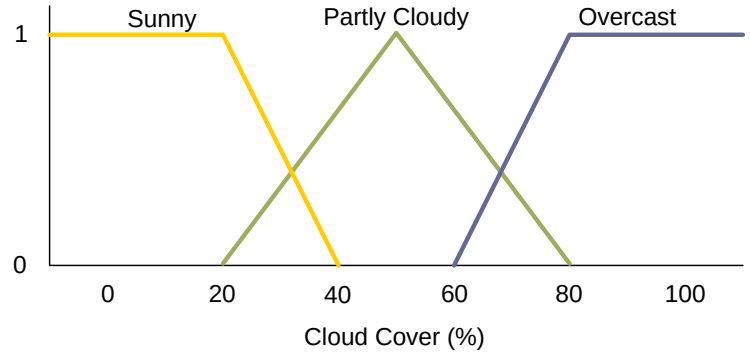


Inputs: Temperature, Cloud Cover

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

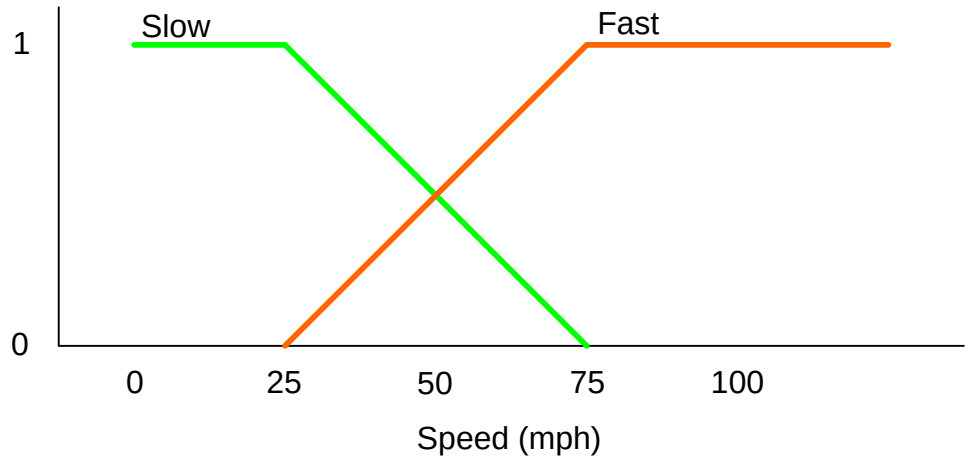


- Cover: {Sunny, Partly, Overcast}



Output: Speed

- Speed: {Slow, Fast}



Rules

- If it's Sunny and Warm, drive Fast
  $Sunny(Cover) \wedge Warm(Temp) \Rightarrow Fast(Speed)$

- If it's Cloudy and Cool, drive Slow
  $Cloudy(Cover) \wedge Cool(Temp) \Rightarrow Slow(Speed)$

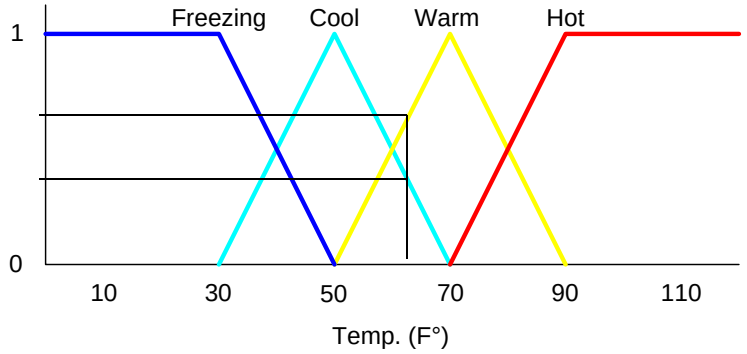
- Driving Speed is the combination of output of these rules...

Example Speed Calculation

- How fast will I go if it is
  - 65 F°
  - 25 % Cloud Cover ?

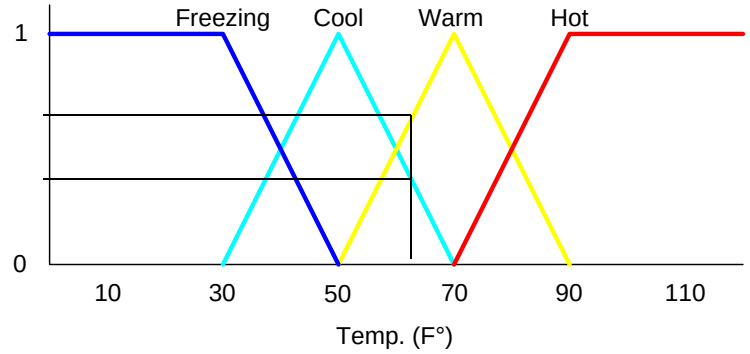
Fuzzification: Calculate Input Membership Levels

- 65 F° $\Rightarrow$ Cool = 0.4, Warm= 0.7

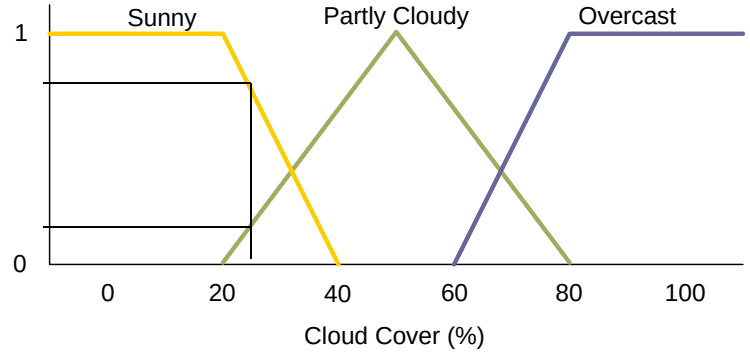


Fuzzification: Calculate Input Membership Levels

- 65 F° ⇒ Cool = 0.4, Warm= 0.7



- 25% Cover ⇒Sunny = 0.8, Cloudy = 0.2



...Calculating...

- If it's Sunny and Warm, drive Fast

$$Sunny(Cover) \wedge Warm(Temp) \Rightarrow Fast(Speed)$$

$$0.8 \wedge 0.7 = 0.7$$

$$\Rightarrow \textbf{Fast = 0.7}$$

- If it's Cloudy and Cool, drive Slow

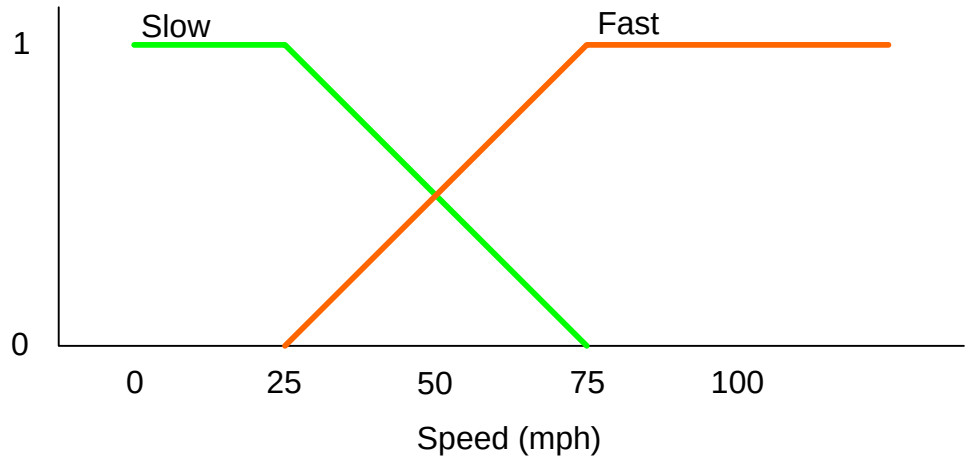
$$Cloudy(Cover) \wedge Cool(Temp) \Rightarrow Slow(Speed)$$

$$0.2 \wedge 0.4 = 0.2$$

$$\Rightarrow \textbf{Slow = 0.2}$$

Defuzzification: Constructing the Output

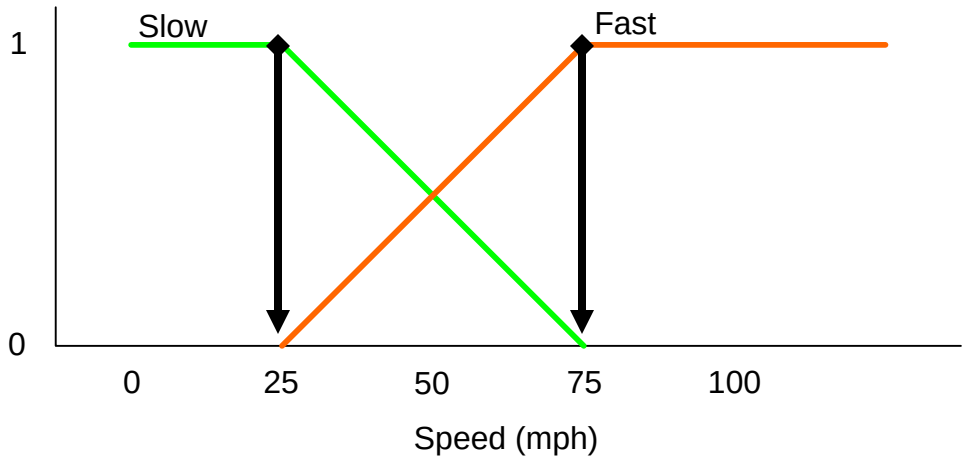
- Speed is 20% Slow and 70% Fast



- Find centroids: Location where membership is 100%

Defuzzification: Constructing the Output

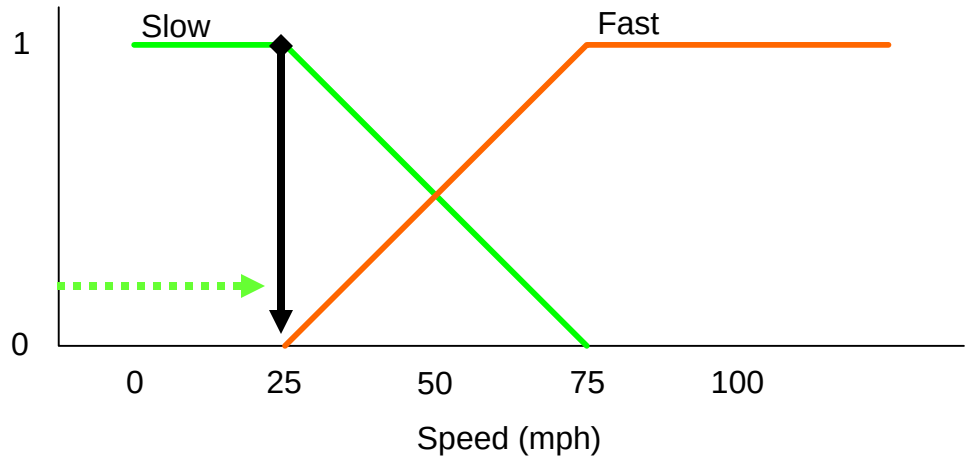
- Speed is 20% Slow and 70% Fast



- Find centroids: Location where membership is 100%

Defuzzification: Constructing the Output

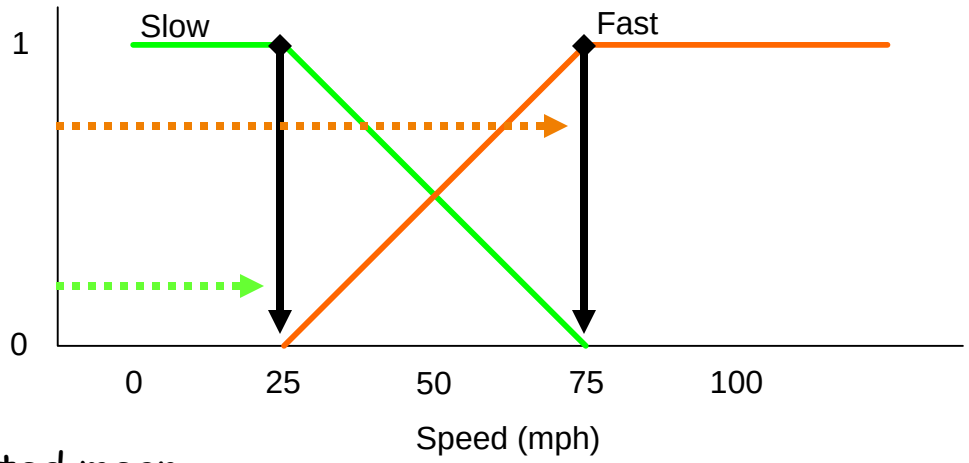
- Speed is 20% Slow and 70% Fast



- Speed = weighted mean
  = (2*25+...

Defuzzification: Constructing the Output

- Speed is 20% Slow and 70% Fast



- Speed = weighted mean
  = (2*25+7*75)/(9)
  = 63.8 mph

Notes: Follow-up Points

- Linguistic variables are used making it similar to how we (humans) think.
- Simplicity in that fuzzy systems do away with complex analytical equations in favor of descriptive rules (possibly from rules of thumb).
- Fast prototyping, cheap, and often quite robust (rule-bases generate non-linear and complex surfaces if viewed as functions).
- 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

Notes: Drawbacks to Fuzzy logic

- Requires tuning of membership functions
- Fuzzy Logic control may not scale well to large or complex problems
- Deals with imprecision, and vagueness, but not uncertainty