GAME AI

with Jeff Wilson, PhD

Fuzzy Logic

Non-Numeric Linguistic Expression

1. Cut two slices of bread **medium thick**.
2. Turn the heat on the griddle on **high**.
3. Grill the slices on one side until **golden brown**.
4. Turn the slices over and add a **generous helping** of cheese.
5. Replace and grill until the top of the cheese is **slightly brown**.
6. Remove, sprinkle on a **small amount** of black pepper, and eat.

Non-Numeric Linguistic Expression

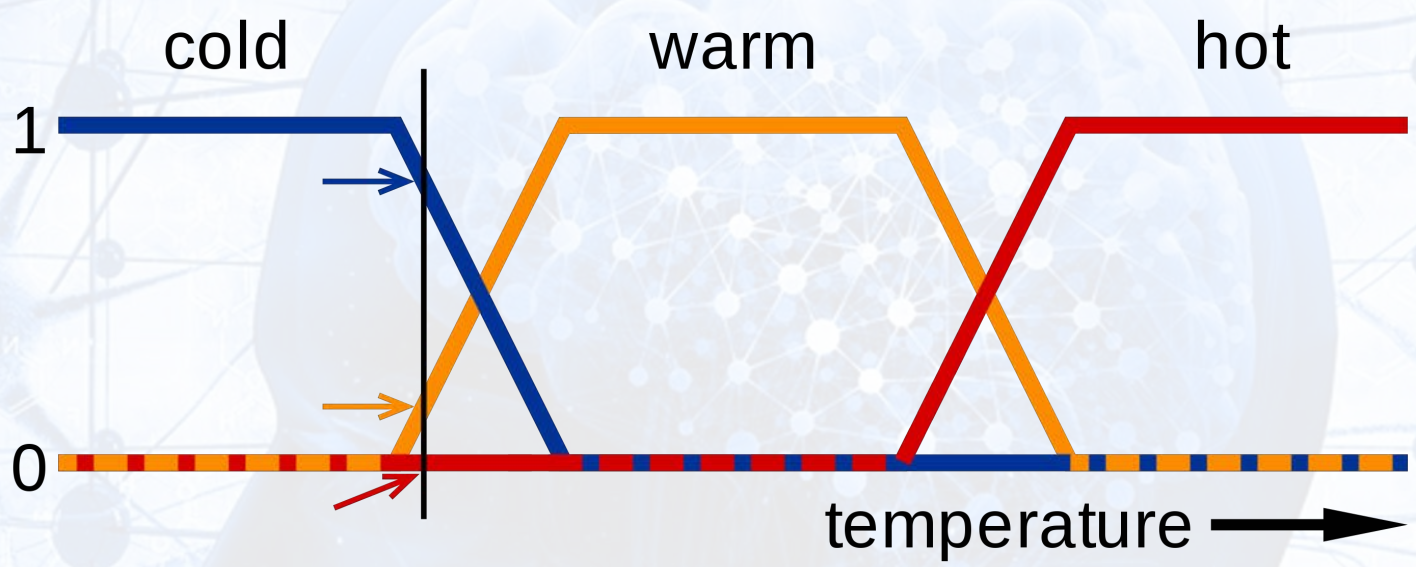
- Subjectivity
- Vagueness
- Flexible interpretation
- Challenge to coordinate with conventional logic

Motivation

- Fuzzy logic: truth degrees, vagueness, subjectivity
  - Consider: “20% chance of rain” vs “partly cloudy”
  - Probability as likelihood, ignorance, uncertainty
- Linguistic: non-numeric expression of rules & facts (e.g., new or old)
  - Allows for use of vague human assessments in computing problems
  - E.g., Cautious vs Confident
- FSM w/ 2 states – switching looks unnatural
  - Cautious (range), sneak slowly (range)
  - Confident, walk normally
- Value?
  - Relatively popular in games industry
  - Largely dismissed in academic AI

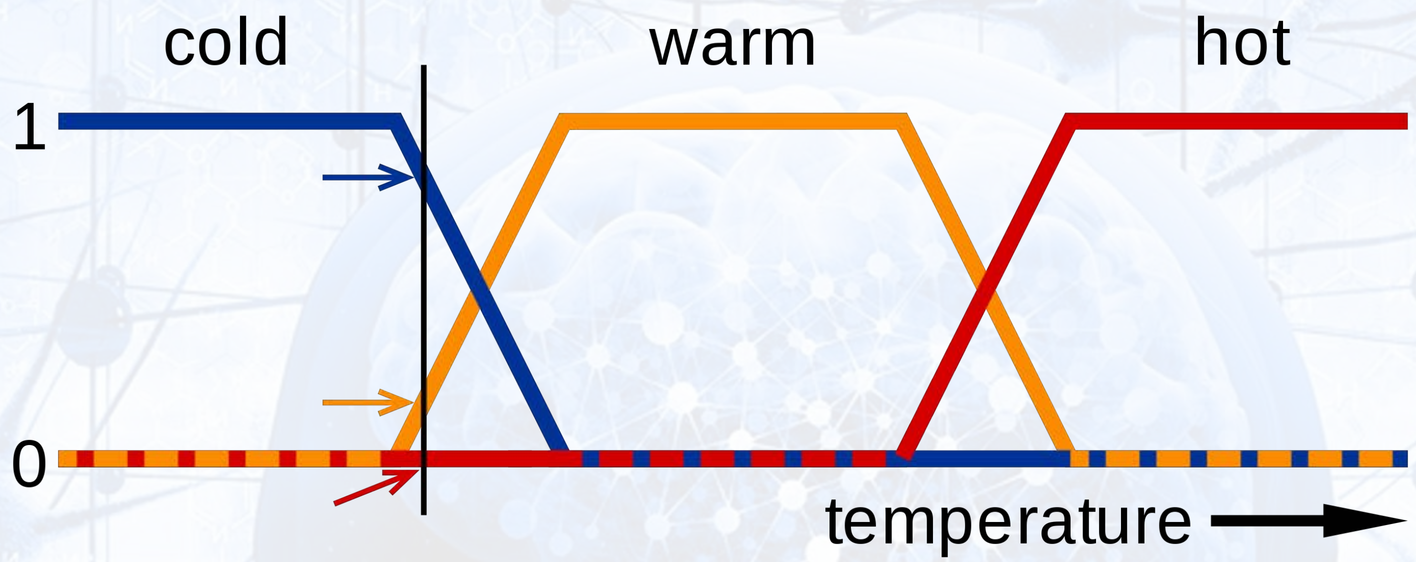
Fuzzy Logic

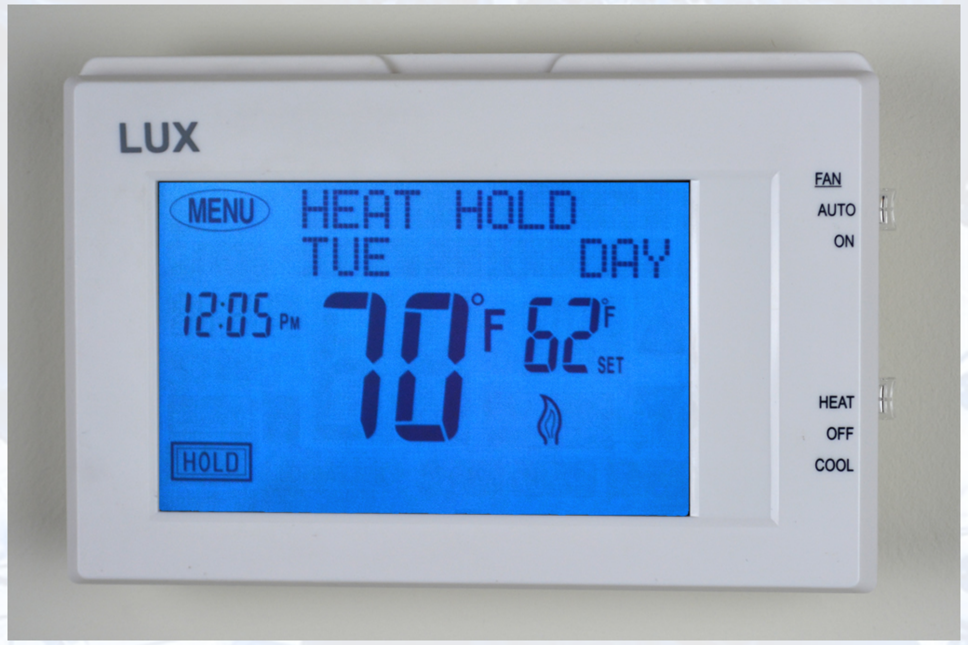
- Modeling of *imprecise concepts*
- Modeling of *imprecise dependencies*
- Superset of classical logic
- Introduces concept of degree of membership (DOM)
- Uses fuzzy sets and fuzzy rules
- **Not** probability



Thermostat Example

- How to adjust the thermostat?
  - How do I feel now?
    - Freezing, Cool, Warm, Hot
  - How much do I adjust the temperature by?
    - Down a lot, down a little, keep same, up, some, up a lot
- **Inputs:**
  - Crisp values (e.g., 75F)
- **Outputs:**
  - Crisp values (turn temp down 5F)
  - Booleans, categories, nearly anything





Thermostat Rules

IF temperature IS very cold THEN fan_speed is stopped

IF temperature IS cold THEN fan_speed is slow

IF temperature IS warm THEN fan_speed is moderate

IF temperature IS hot THEN fan_speed is high

Fuzzy Logic Use Cases

- More direct way to represent expertise (linguistic form)
  - Potential for declarative authoring by non-programmers
- Opportunity to capture expertise (e.g., consult an expert) but lack the resources to train (Neural Net, etc.)
- Flexible behavior design paradigm
- When behavior design efficiency is more important than optimization
- Behaviors can act on multiple variables without imposing rigid structure to decision making

Fuzzy Logic History

- Proposal of Fuzzy Set Theory Introduced in 1965 by Lotfi A. Zadeh (UC Berkeley)
- Japanese were first to utilize for commercial applications in late 1970s-1980s (high-speed train, rice cookers)
- Use of Fuzzy Logic controllers really picked up late 1980s
- Research boom in ‘90s

Golfing Game Example

- When putting: *If the **ball is far from the hole** and the **green is sloping gently downward from left to right**, then **hit the ball firmly** and at an **angle slightly to the left of the flag**.*
- When putting: *If the **ball is very close to the hole**, and the **green between the ball and hole is level**, then **hit the ball gently** and **directly at the hole**.*
- When driving from the tee: *If the **wind is of strong force** and **blowing right to left**, and the **hole is far away**, then **hit the ball hard** and at an **angle far to the right of the flag**.*

Golfing Game Example

- Close = the ball is between 0 meters and 2 meters from the hole.
- Medium = the ball is between 2 meters and 5 meters from the hole.
- Far = the ball is greater than 5 meters from the hole.
- **Problem**: What if the ball is 4.99 meters away?
  - Transition from medium to far is abrupt with significant impact on behavior

Golfing Game Example

When driving from the tee: *If the **wind is of strong force** and **blowing right to left**, and the **hole is far away**, then **hit the ball hard** and at an **angle far to the right of the flag**.*

```
If(abs(wind) >= MIN_STRONG_WIND AND wind < 0 AND dist(ball.pos, hole.pos) >=
MIN_FAR) THEN
        hitForce = lerp(MIN_STRONG_FORCE, MAX_STRONG_FORCE,
                        inverseLerp(MIN_STRONG, MAX_STRONG, abs(wind)))
        Angle = angle(ball.pos, hole.pos) + lerp(MIN_ANGLE, MAX_ANGLE,
                ((inverseLerp(MIN_STRONG, MAX_STRONG, abs(wind)) +
                inverseLerp(MIN_FAR, MAX_FAR, dist(ball.pos, hole.pos))*0.5))
```

Boolean Logic

- Predicates return true or false
  - *IsArmed(agent), IsInjured(agent)*
- Set membership: one or the other

Fuzzy Sets

- Degree of Membership (DOM)
- Instead of predicates being true or false, there is a normalized value specifying degree of membership
- **Example**: *IsInjured(agent) == 0.7*
- The value is **not** a probability or a percentage, it is only the degree to which the the *agent* is a member of *IsInjured*

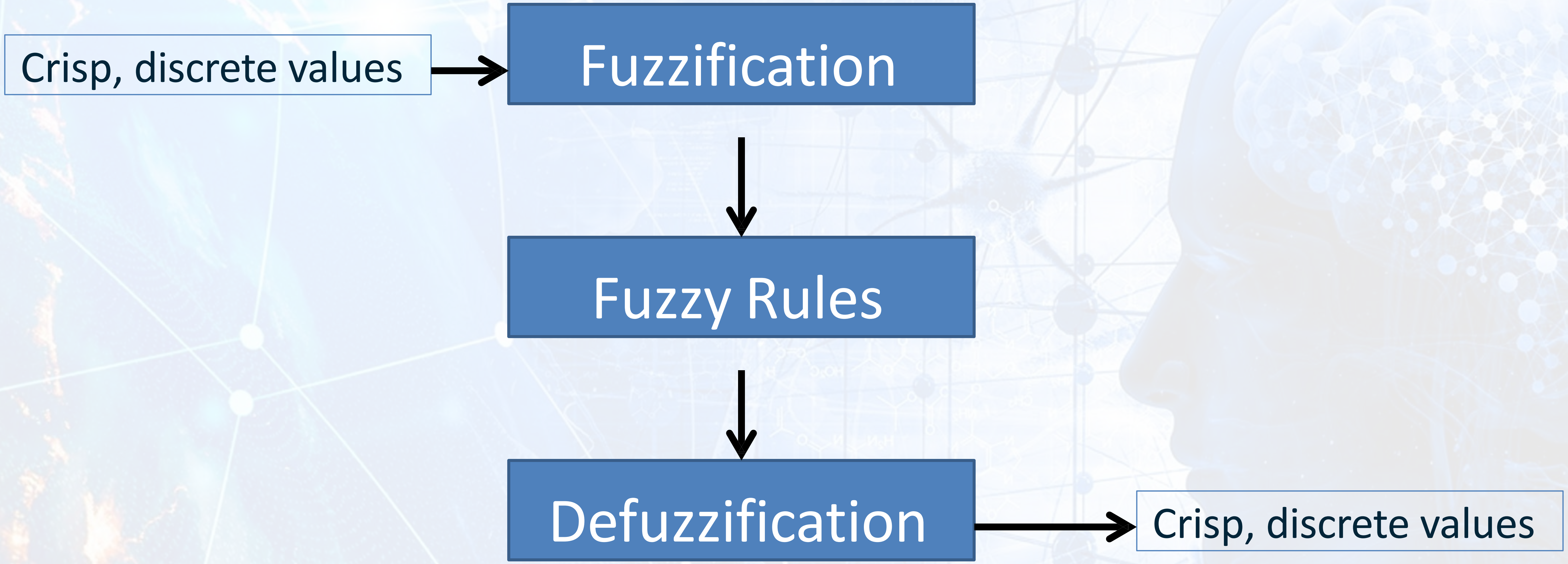
Fuzzy Weirdness

- Some traits that one might expect to be mutually exclusive can simultaneously present non-zero degrees of membership
- **Example**:
  - *IsHealthy(agent) == DOM(0.3)*
  - *IsInjured(agent) == DOM(0.7)*

Degree of Membership

- When multiple DOMs are non-zero, it makes sense that DOM should sum to 1.0
- However, a weighted average is often sufficient for good results

Fuzzy Rules


Crisp, discrete values
Fuzzification
Fuzzy Rules
Defuzzification
Crisp, discrete values

Fuzzy Rules

Crisp, discrete values → Fuzzification

Fuzzy boundaries (partial membership) described with "Membership Functions"

↓

Fuzzy Rules

↓

Defuzzification → Crisp, discrete values

Fuzzy Rules

Crisp, discrete values → Fuzzification

↓

Fuzzy Rules

IF *antecedent* THEN *consequent*
(where antecedent has DOM,
and consequent fires by degree)

↓

Defuzzification → Crisp, discrete values

Fuzzy Rules

Crisp, discrete values → Fuzzification

↓

Fuzzy Rules

↓

Defuzzification → Crisp, discrete values

Combines consequents fired into a crisp value.

Fuzzy Inference

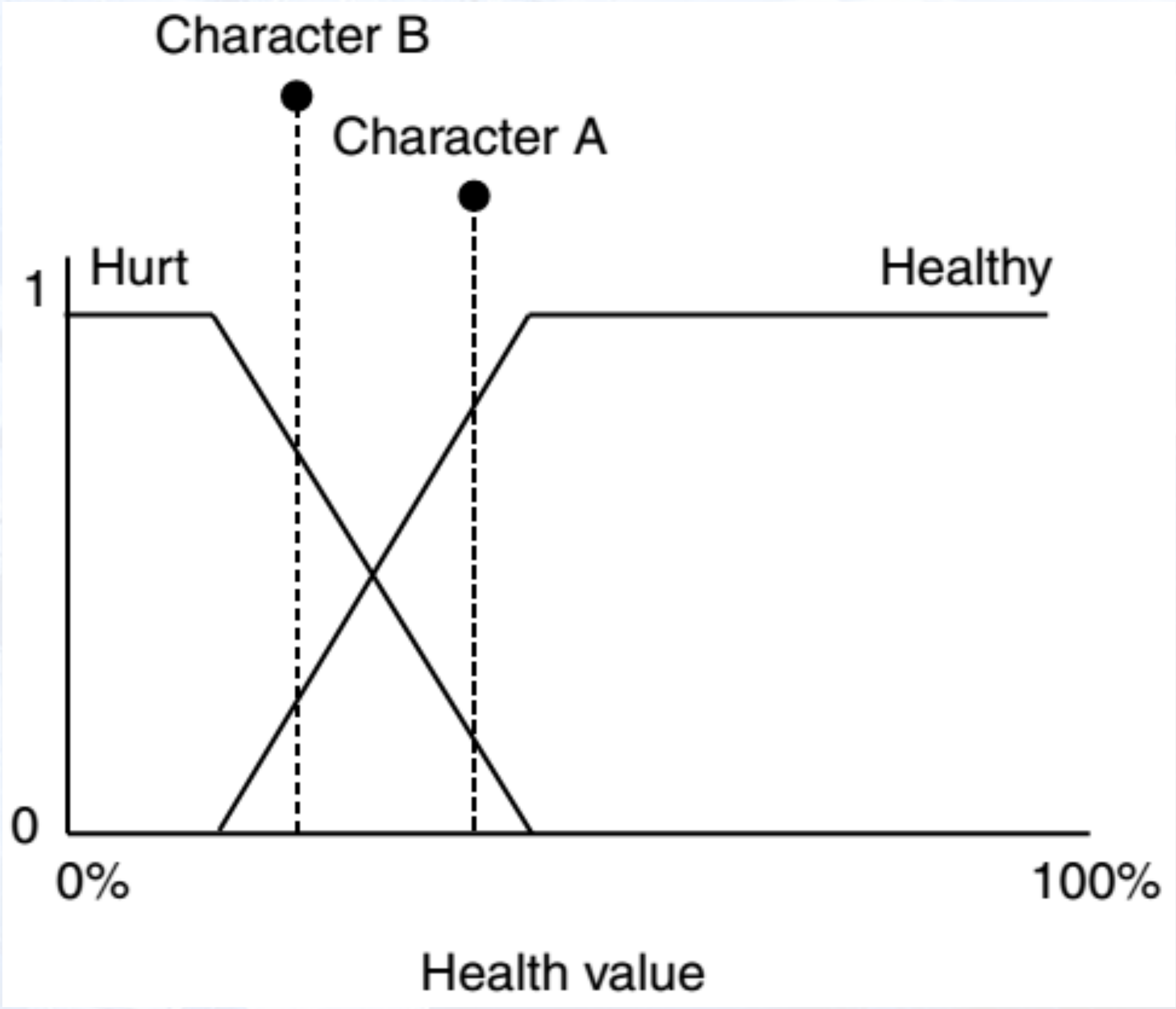
- For each rule,
  - For each antecedent, calculate the degree of membership of the input data.
  - Calculate the rule's inferred conclusion based upon the values in previous step
- Combine all the inferred conclusions into a single conclusion (a fuzzy set)
- For crisp values, the conclusion from 2 must be defuzzified

Fuzzy Rules

- Relate the known membership of certain fuzzy sets to generate new DOM values for other (output) fuzzy sets
  - Must create rule for each possible combination of antecedent sets
- E.g., “If I am close to the corner AND I am traveling fast, then I should brake”
- $m_{\text{(should brake)}} = \min(m_{\text{(close to corner)}}, m_{\text{(traveling quickly)}})$
- Membership of should break with “close to corner” 0.6 and “traveling fast” 0.9?

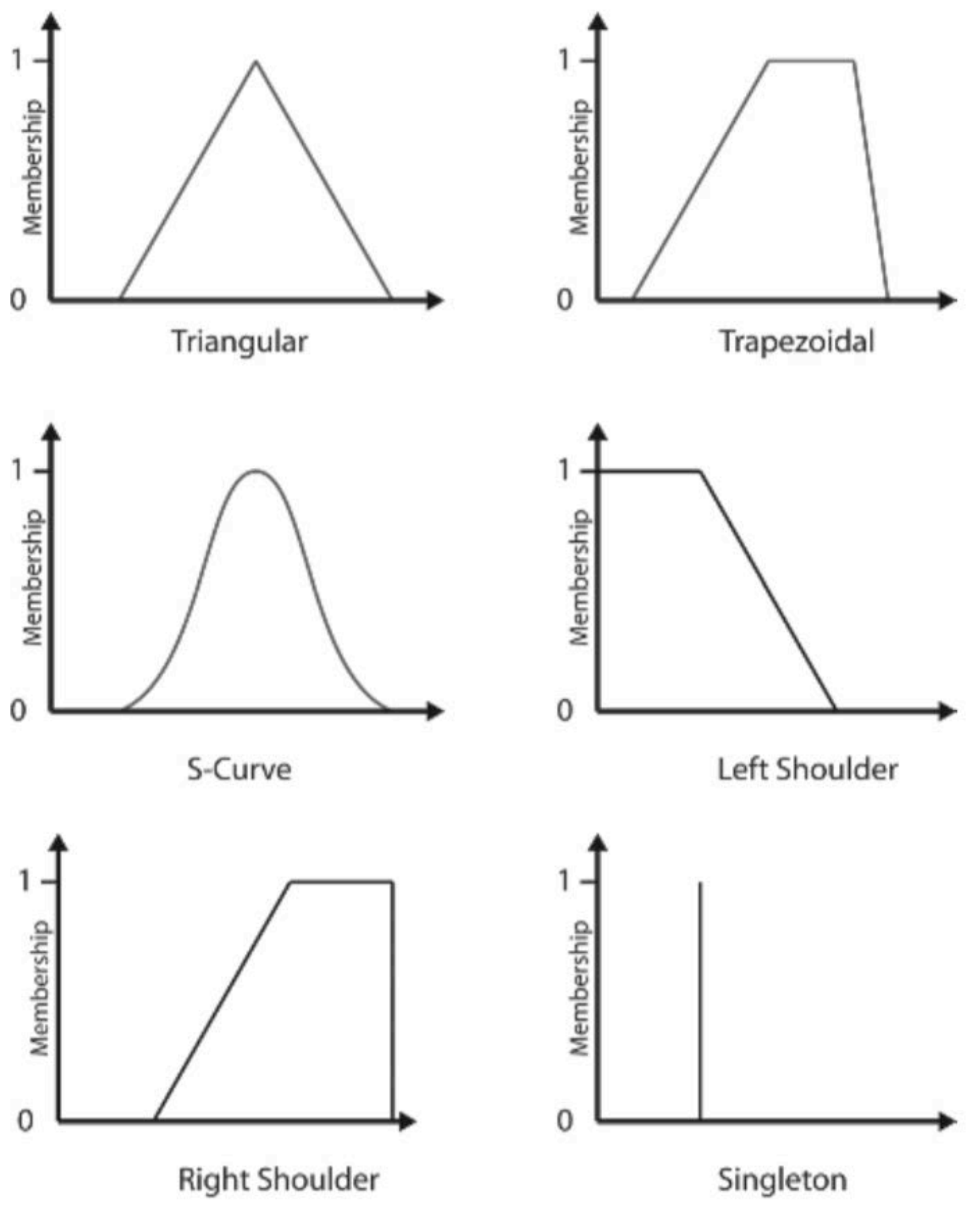
Fuzzification

- Fuzzification:
  Game state → DOM "Membership Function"
  - Triangular
  - Trapezoidal
  - S-Curve
  - Left/Right Shoulder
  - Singleton
- Note: Vert. lines *should* sum to 1
- De-fuzz:
  DOM → Game state
- Numeric Fuzzification:
  f(numeric) → DOM



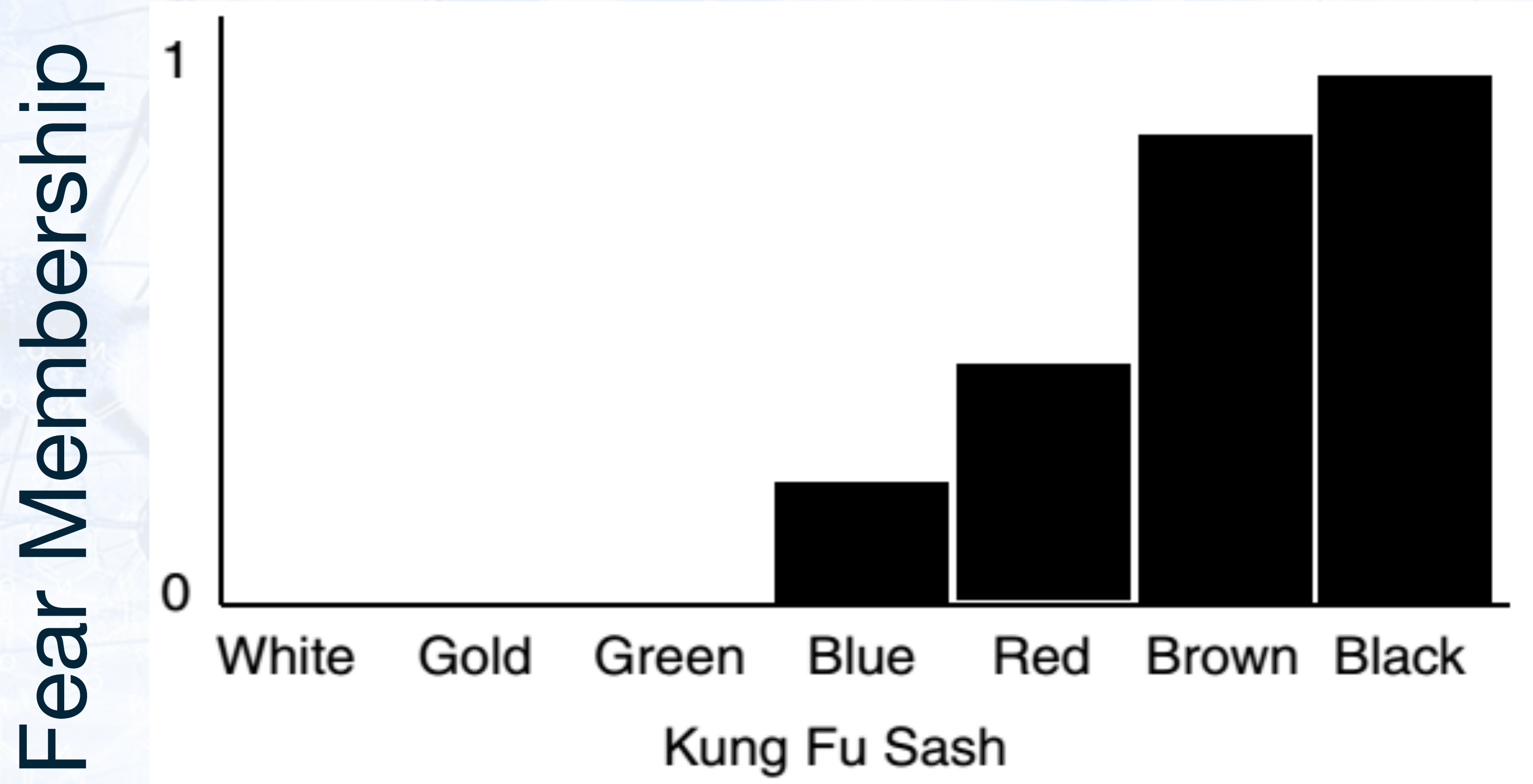
F(health val) → DOM(healthy)
F(health val) → DOM(hurt)

Example membership functions



Fuzzification of Small Sets and Enumerated Types

- Store pre-determined membership values. E.g. kung fu game
- (fuzz. func. is a lookup table)
  - Boolean var
    - hasPwrflArtifact
  - Enum var
    - DOM of fearsmFighter from one of set of sashes



Set Operations – AND/OR/NOT

Boolean logic: True, False

Fuzzy: DOM of a fuzzy set

Little rain (0.3) AND very cold (0.8)

Fuzzy Logic

$m_{(A \&\& B)} = \min(m_A, m_B)$

$m_{(A\ OR\ B)} = \max(m_A, m_B)$

$m_{(NOT\ A)} = 1 - m_A$

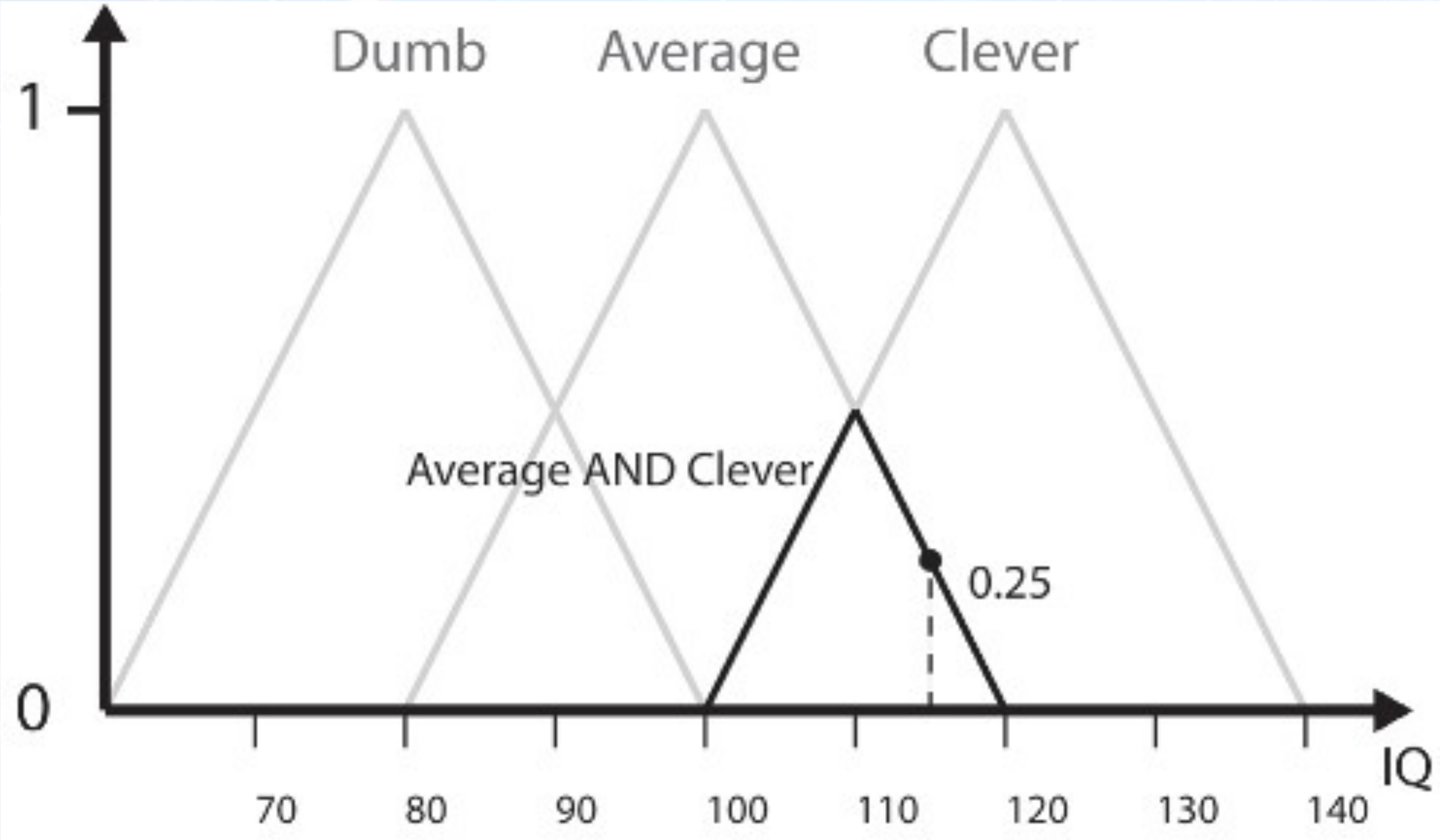
Hedge: VERY = $(m_A)^2$

Hedge: FAIRLY = $(m_A)^{0.5}$

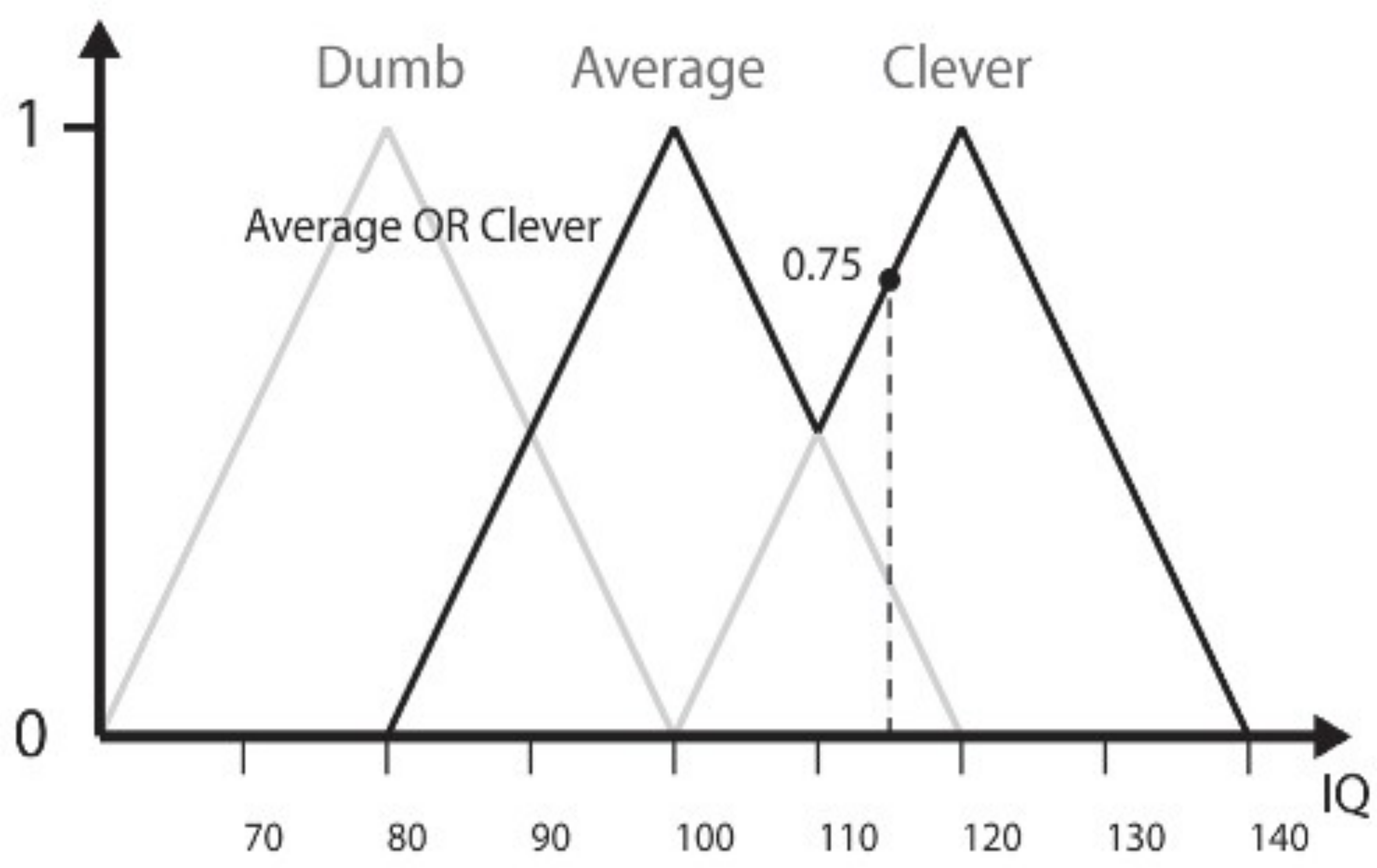
A	B	A && B	A OR B
F	F	F	F
F	T	F	T
T	F	F	T
T	T	T	T

Visualize Fuzzy Logic

$m_{(A \&\& B)} = \min(m_A, m_B)$



$m_{(A \text{ OR } B)} = \max(m_A, m_B)$



Alternate Fuzzy Logical Expressions

- x AND y = x*y
- x OR y = 1-(1-x)*(1-y) = x+y-x*y

Still use:

$m_{(NOT\ A)} = 1 - m_A$

- Derivation of x OR y:

  x OR y = NOT(AND(NOT(x), NOT(y)))

  x OR y = NOT(AND(1-x, 1-y))

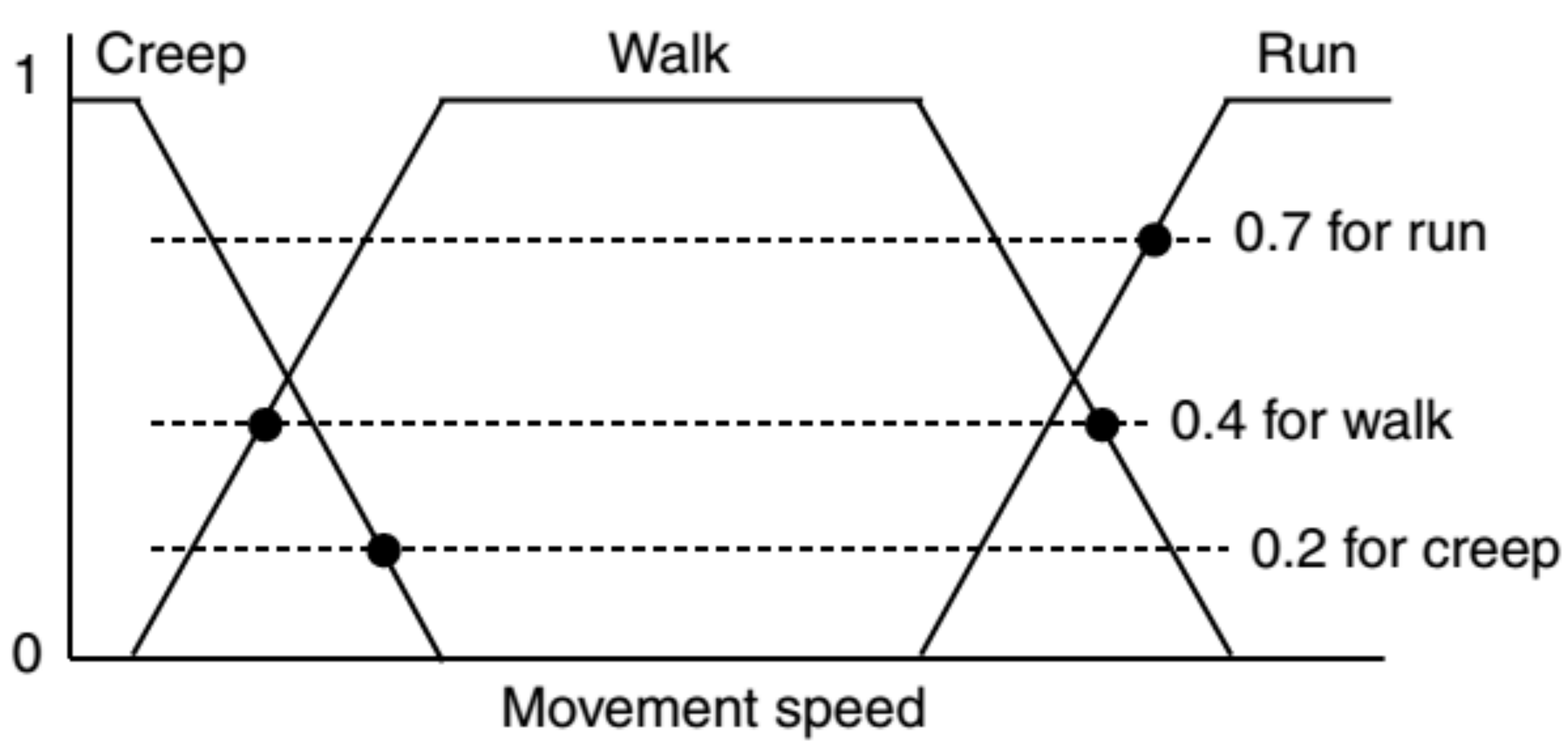
  x OR y = NOT((1-x)*(1-y))

  x OR y = 1-(1-x)*(1-y)

Defuzzification

- Need to translate data back after applying whatever logic was needed
- Multiple approaches
  - Mean of maximum
  - Centroid
  - Average of Maxima
  - …
- **Problem**: Turn a set of membership values into a (typically) single number (crisp value)

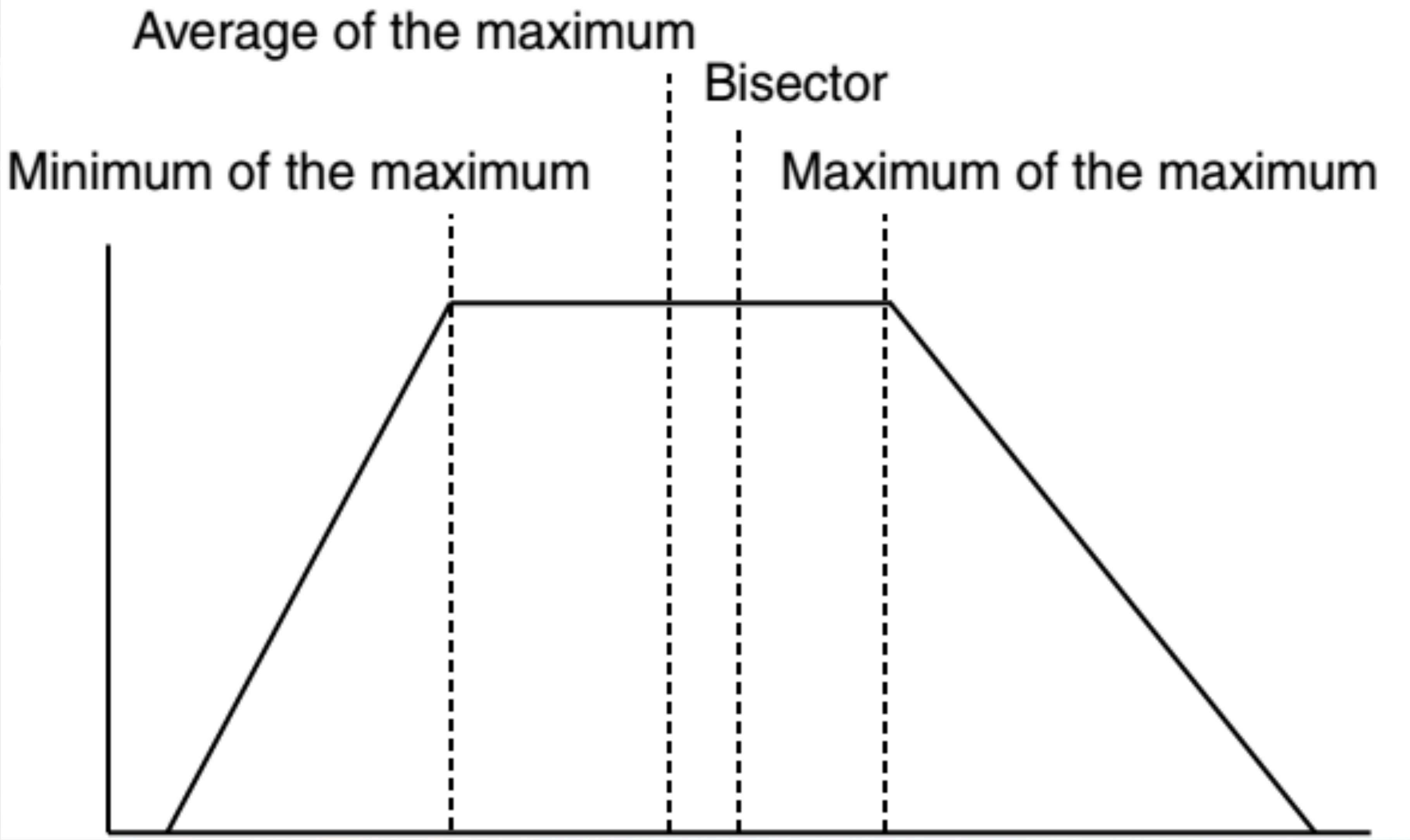
Defuzzification



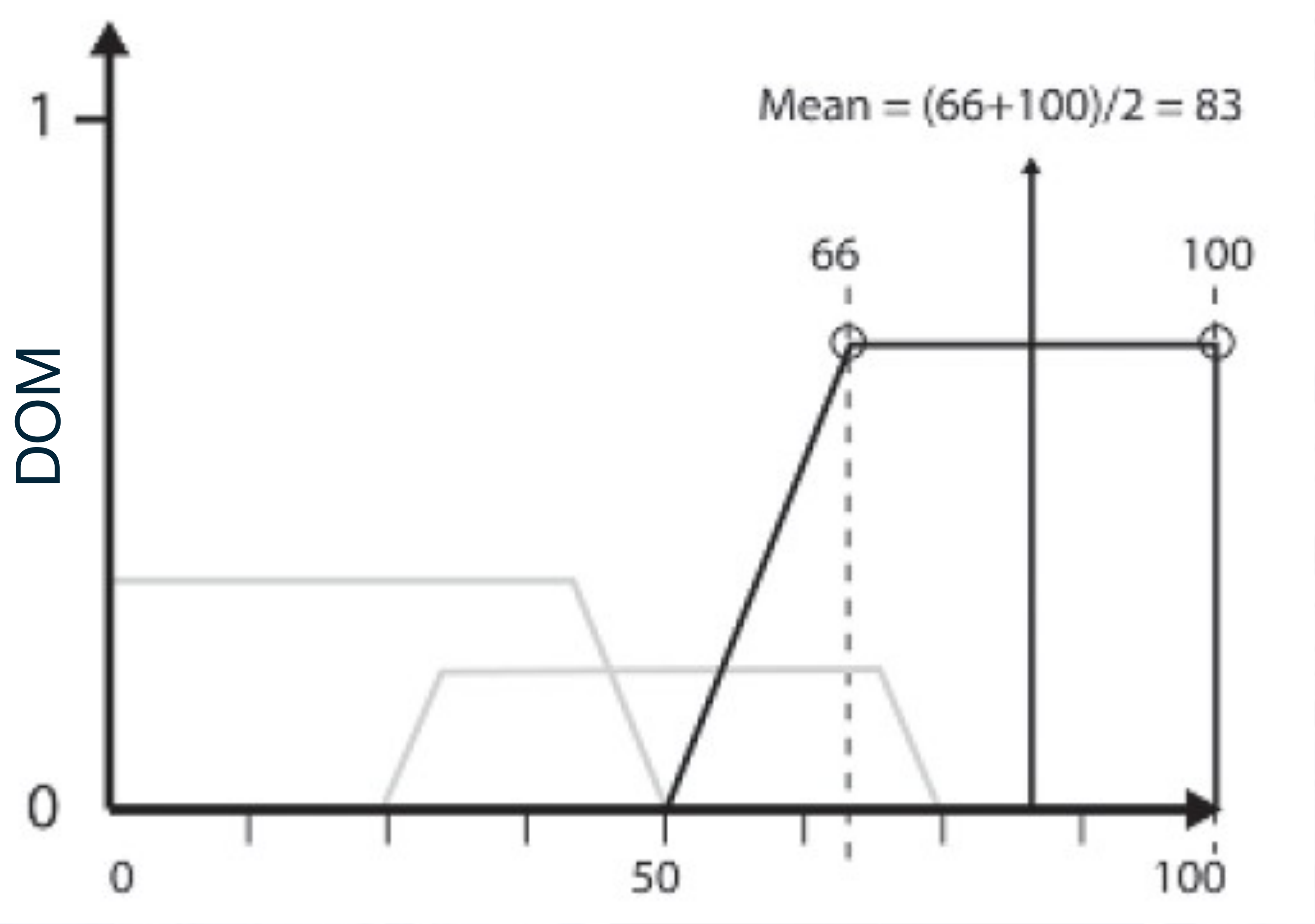
Fuzzy Set includes all 3

Note that it is possible for a DOM line to intersect membership function more than once

Highest Membership



Mean of the Maximum

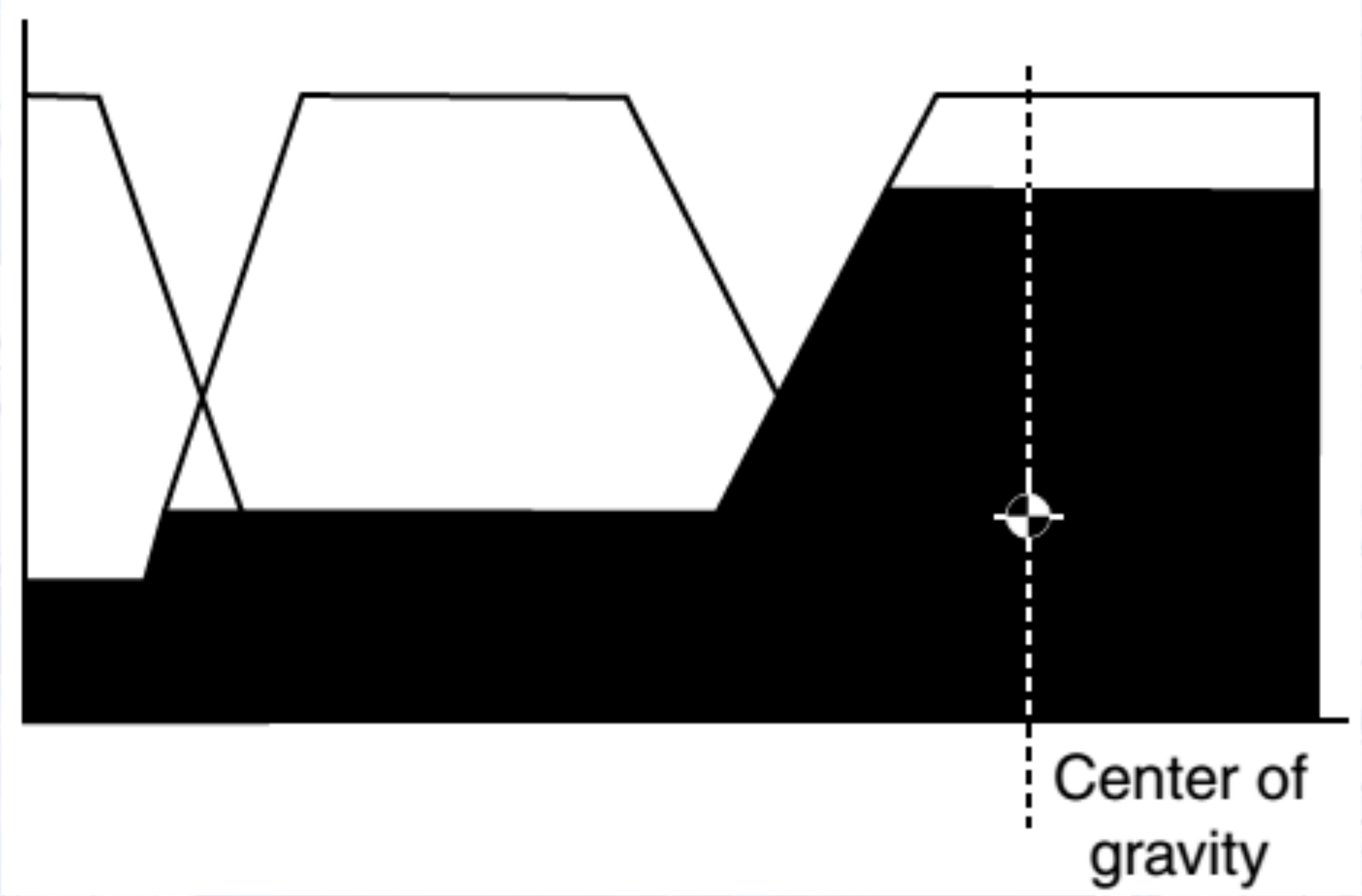


Blend based on Membership

- Use DOM as weights
  - 0.33 creep, 0.33 walk, 0.34 run
  - 0.33 * characteristic creep speed + 0.33 * characteristic walk speed + 0.34 * characteristic run speed
- Normalize values
  - if not already guaranteed to sum to 1.0
- Can use minimum values (Smallest of Maximum method or Left of Maximum, LM)

Center of Gravity

- Crop membership function at
- DOM value
- Integrate each in turn to find center of gravity
- Method often used, but is expensive
- Blending works about as well and is cheap



Buckland’s Rocket Launcher Example

(distance = 200 and ammo status = 8)

Rule 1. IF Target_Far AND Ammo_Loads THEN Desirable

Rule 2. IF Target_Far AND Ammo_Okay THEN Undesirable

Rule 3. IF Target_Far AND Ammo_Low THEN Undesirable

Rule 4. IF Target_Medium AND Ammo_Loads THEN VeryDesirable

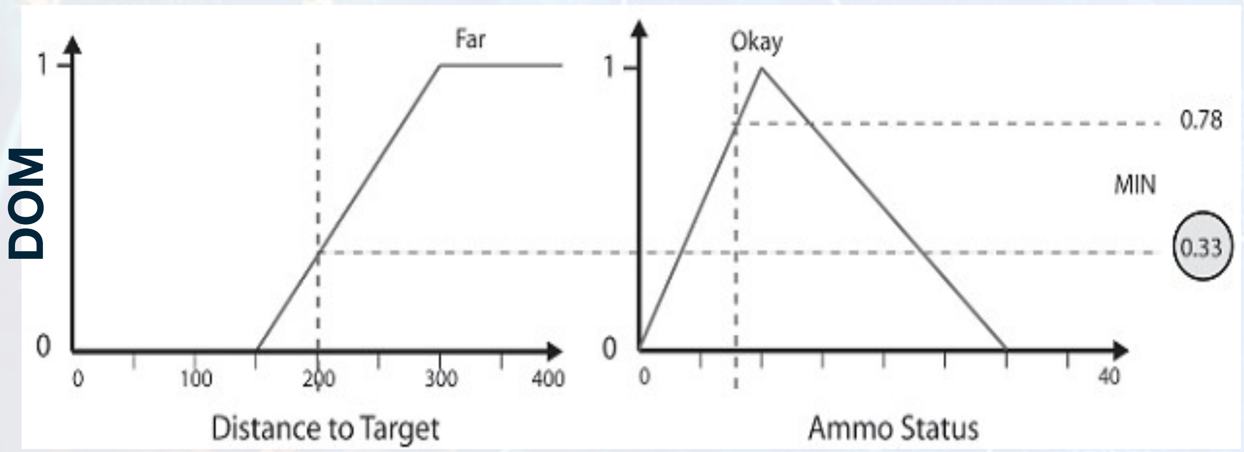
Rule 5. IF Target_Medium AND Ammo_Okay THEN VeryDesirable

Rule 6. IF Target_Medium AND Ammo_Low THEN Desirable

Rule 7. IF Target_Close AND Ammo_Loads THEN Undesirable

Rule 8. IF Target_Close AND Ammo_Okay THEN Undesirable

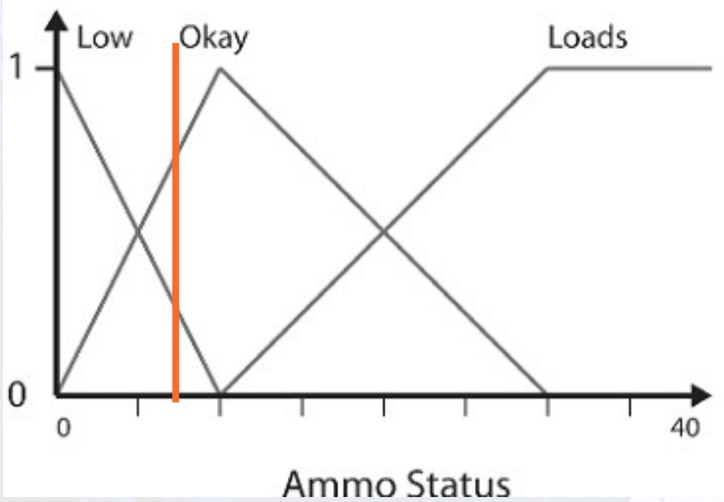
Rule 9. IF Target_Close AND Ammo_Low THEN Undesirable

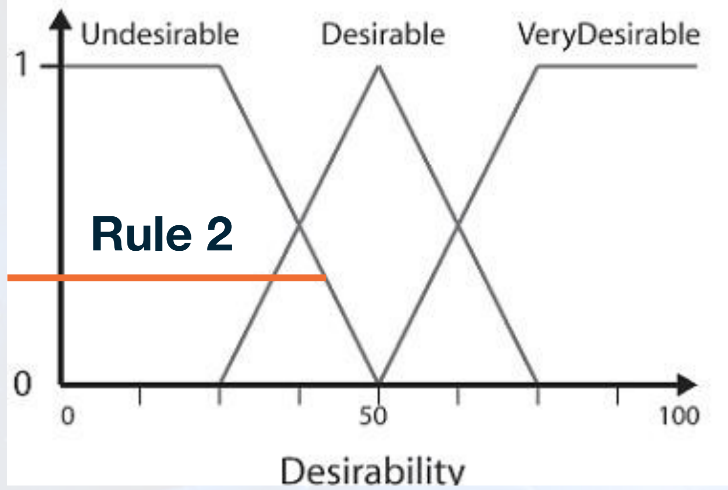


Fuzzy associative matrix

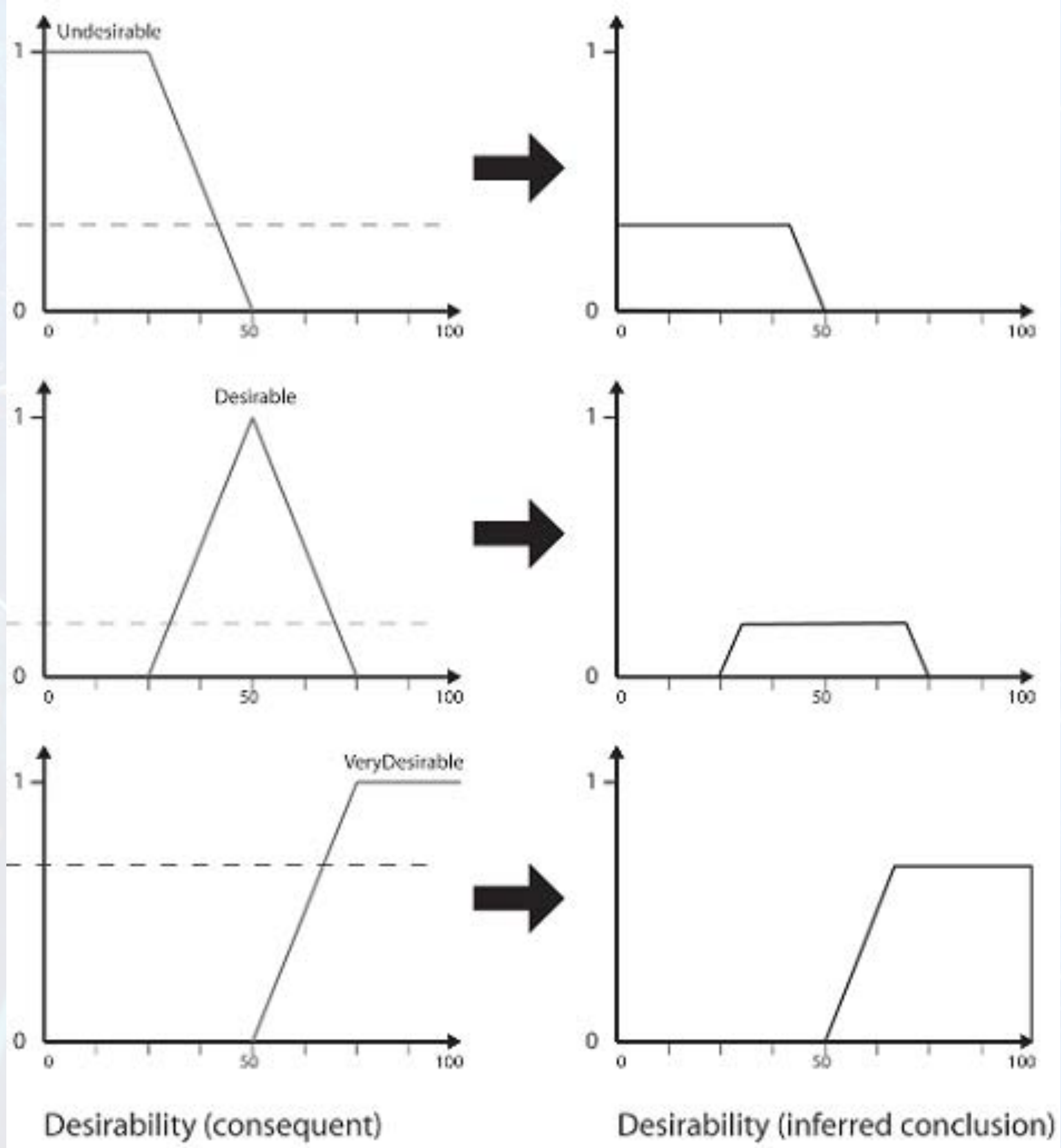
	Target_Close	Target_Medium	Target_Far
Ammo_Low	Undesirable 0	Desirable 0.2	Undesirable 0.2
Ammo_Okay	Undesirable 0	VeryDesirable 0.67	Undesirable 0.33
Ammo_Loads	Undesirable 0	VeryDesirable 0	Desirable 0

Consequent	Confidence
Undesirable	0.33
Desirable	0.2
Very Desirable	0.67

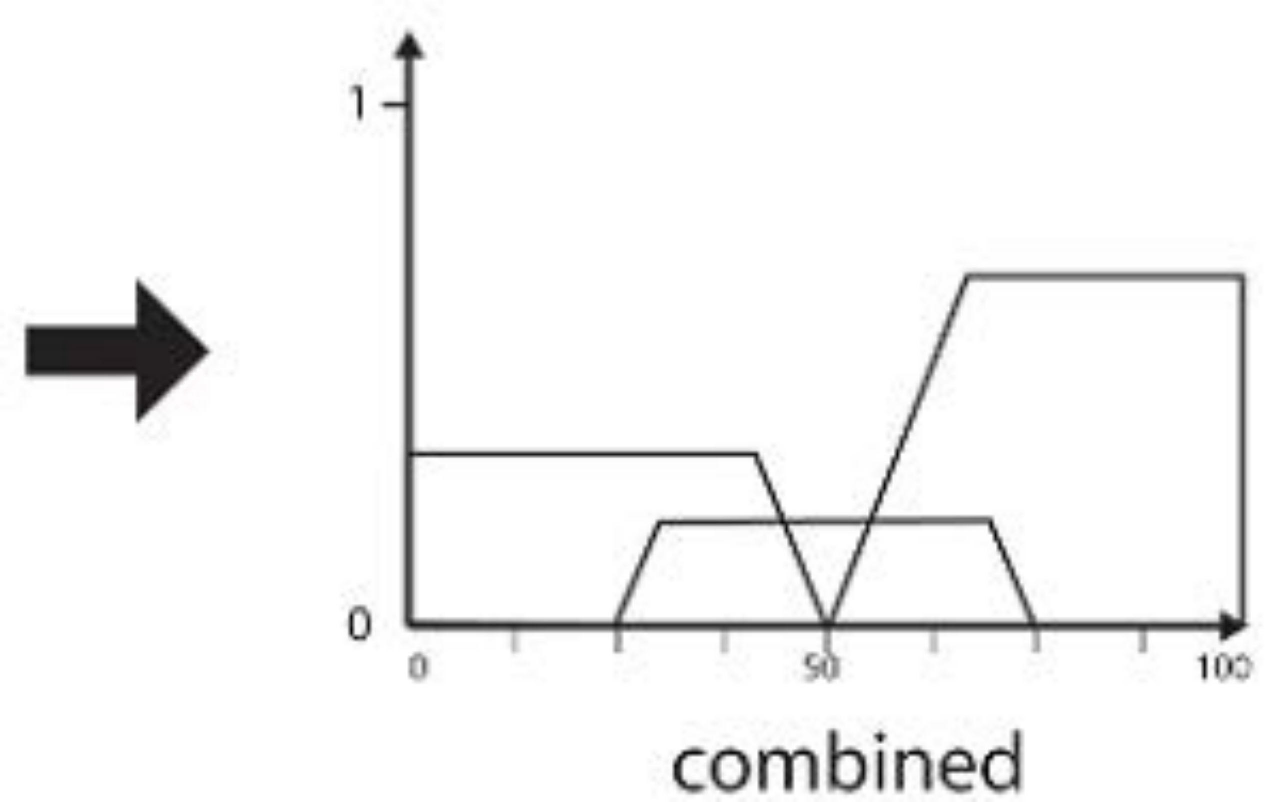




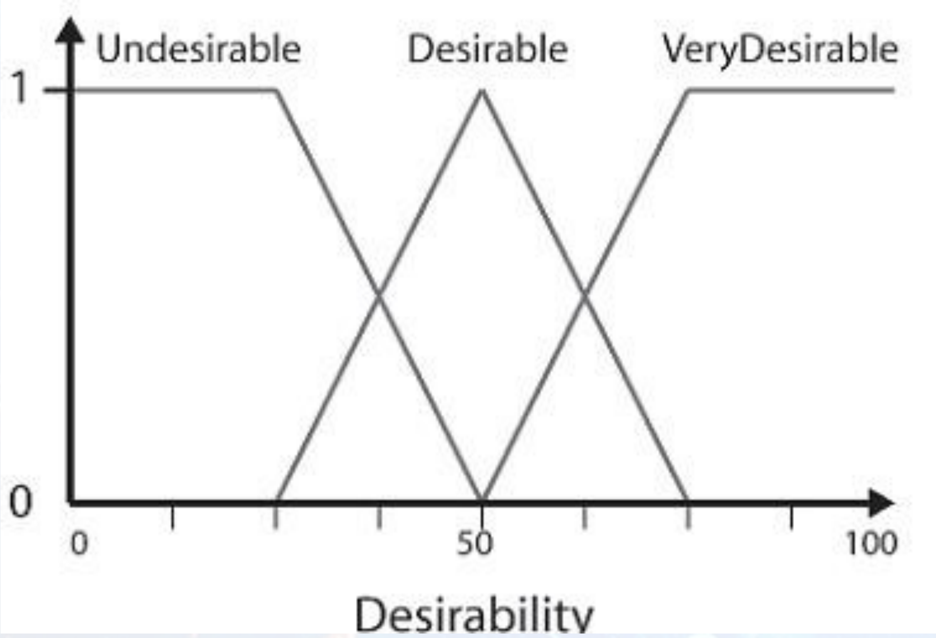
Buckland's Rocket Launcher Example

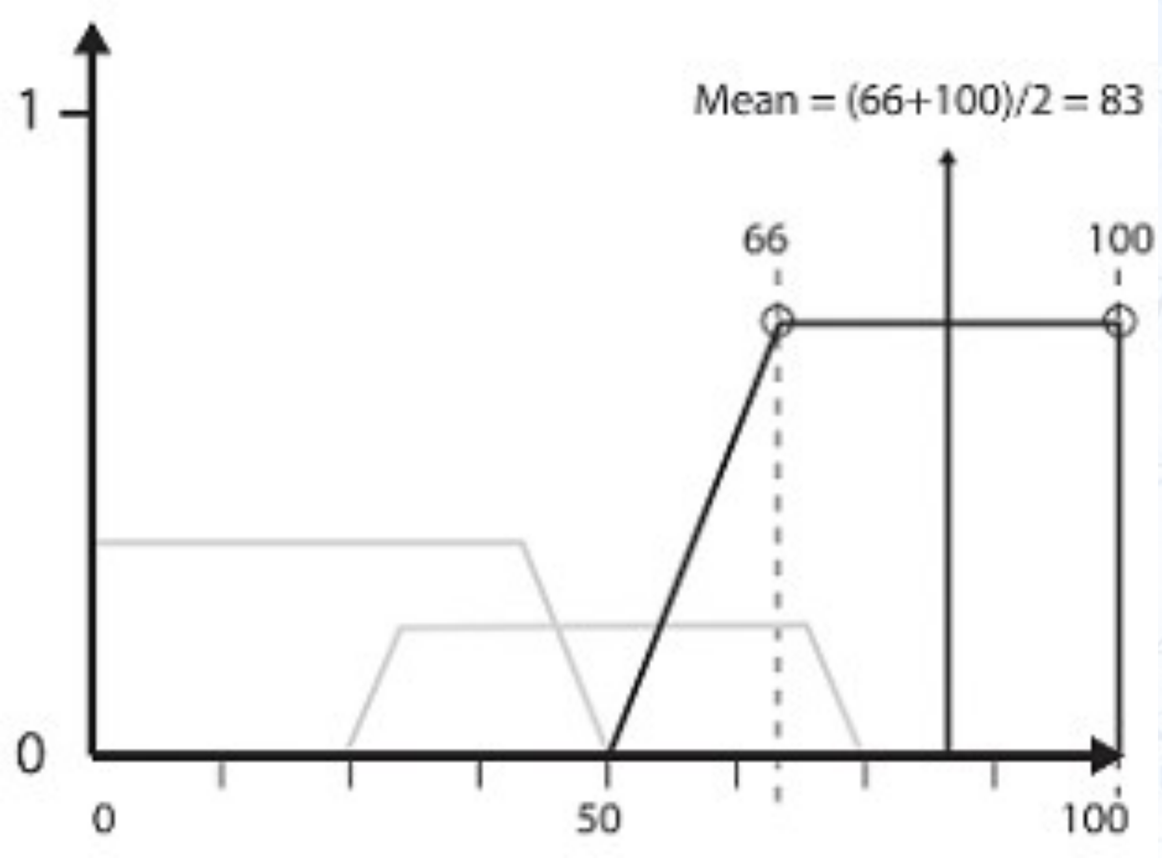


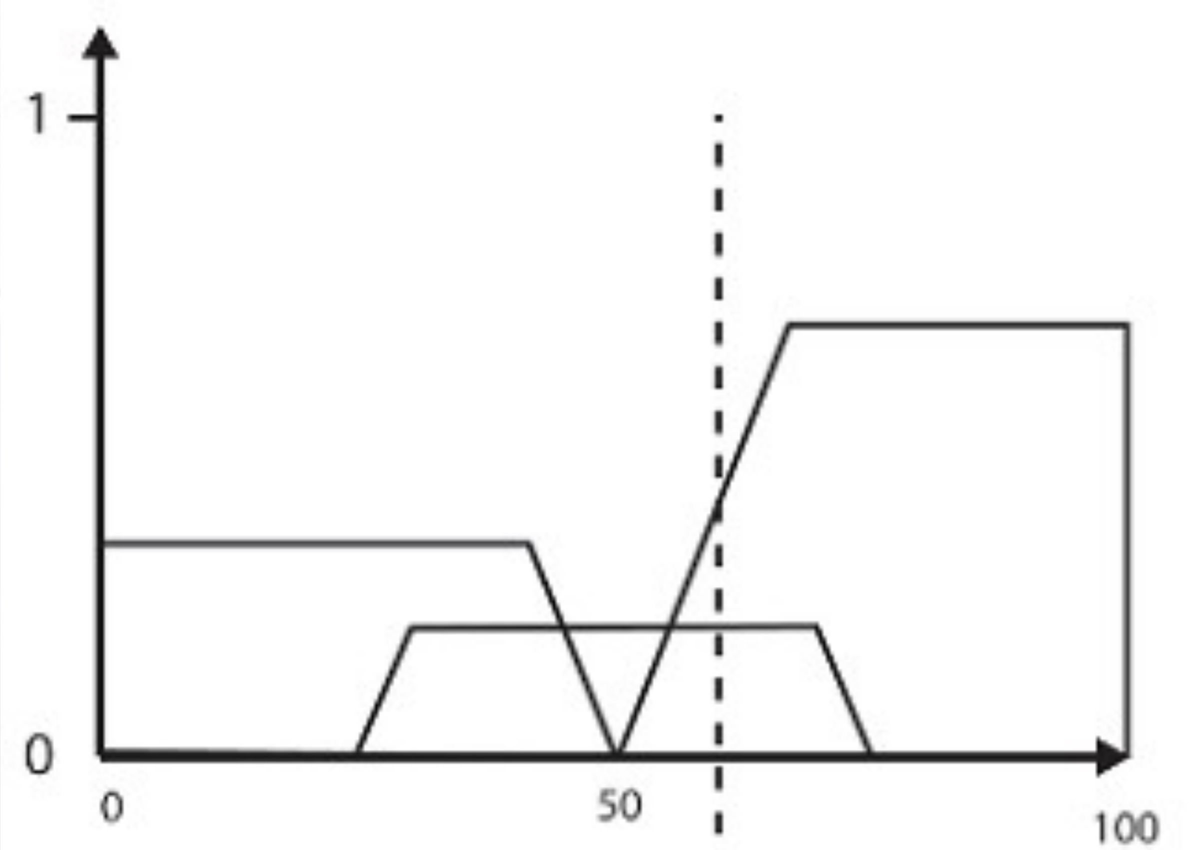
Consequent	Confidence
Undesirable	0.33
Desirable	0.2
Very Desirable	0.67

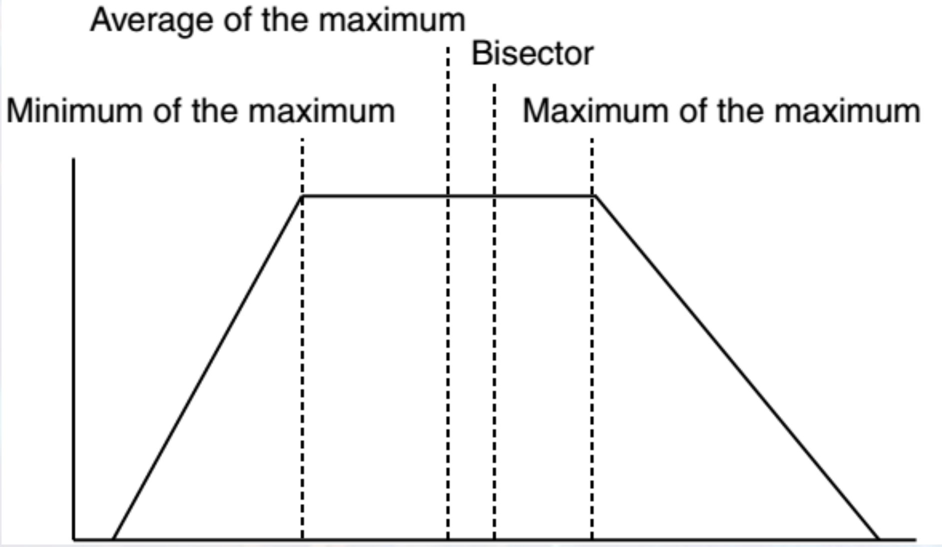


Buckland's Rocket Launcher Example









Set	Representative Value	Confidence
Undesirable	12.5	0.33
Desirable	50	0.2
VeryDesirable	87.5	0.67

$$Desirability = \frac{12.5 \times 0.33 + 50 \times 0.2 + 87.5 \times 0.67}{0.33 + 0.2 + 0.67}$$

$$= \frac{72.75}{1.2}$$

$$Desirability = 60.625$$

Fuzzy Decision Making

- Use Fuzzy Logic to select actions/behaviors/states
- Each action can be applied to varying degrees, rather than selected discretely
- Allows blending of actions in gray areas where one action or another is not clear

Fuzzy Rules

- Fuzzy logic can be applied to a decision-making framework that is similar to Rule Based Systems
- There is no formal name for this and may be referred to as Fuzzy State Machines or simply Fuzzy Logic
- Fuzzy State Machines is also used to describe another algorithm, so it is a bit confusing
  - (See Millington for Fuzzy State Machines similar to Finite State Machines but with fuzzy attributes)
- We will simply refer to the concept as Fuzzy Rules

Fuzzy Rules

- Decisions made on some number of crisp inputs
- Each input is mapped to fuzzy states as previously presented
- May require that sum of DOMs from overlapping membership functions sums to 1.0 (allows optimization)
- Output states are fuzzy states representing agent actions
- Fuzzy Rule Structure:
  - IF input-state-1 AND … AND input-state-n THEN output-state
- Example:
  - IF chasing AND corner-entry AND going-fast THEN brake
  - IF leading AND mid-corner AND going-slow THEN accelerate

Fuzzy Rule Structure

- Each clause in a rule is a state from a different crisp input
- Clauses are combined with fuzzy AND
- Common to require all fuzzy input state combinations to be represented
- Each fuzzy output state will be the maximum observed from activated rules

Fuzzy Rules Example

- **Input DOM**
  - Corner-entry = 0.1
  - Corner-exit = 0.9
  - Going-fast = 0.4
  - Going-slow = 0.6
- **Rules**
  - IF corner-entry (0.1) AND going-fast (0.4) THEN brake (min(0.1, 0.4)=0.1)
    IF corner-exit (0.9) AND going-fast (0.4) THEN acc. (min(0.9, 0.4)=0.4)
  - IF corner-entry (0.1) AND going-slow (0.6) THEN acc. (min(0.1, 0.6)=0.1)
  - IF corner-exit (0.9) AND going-slow (0.6) THEN acc. (min(0.9, 0.6)=0.6)

Fuzzy Rules

- If the DOMs for fuzzy input states always sum to 1.0 for any given crisp value, then an optimization can be made:
- Short circuit logic can be employed that recognizes when it is not possible to increase a fuzzy output state due to taking the Min() (Fuzzy AND) of fuzzy input states.
- This allows some rules to be abandoned before all fuzzy input state membership functions are evaluated

Fuzzy Rules Performance

- $O(n+m)$ - for algorithm memory, where *n* is the number of input states and *m* is the number of output states
- Rules also need to be stored:
  - $O(\prod_{k=0}^{i} n_k)$ - where n_k number of states per variable and *i* is the number of input variables
- For Time:
  - $O(i\prod_{k=0}^{i} n_k)$
- Performance is a huge weakness!

Solution: Comb's Method

- Manage complexity of Fuzzy Rules by simplifying them
- IF a-1 AND ... AND a-n THEN c
- Rewritten as:
- IF a-1 THEN c
- ...
- IF a-n THEN c
- Drastically reduces complexity ($O(in)$ - time) but cannot represent all logic possibilities
- Typically, Comb's method is used from the start of design
- In practice, Comb's method can be quite capable because states can interact with one another via defuzzification stage

$$((p \wedge q) \Rightarrow r) \iff ((p \Rightarrow r) \vee (q \Rightarrow r))$$

((p AND q) implies r) EQUIVALENT_TO
((p implies r) OR (q implies r))

Fuzzy Logic Pros and Cons

- **Pros**
  - Easy to understand
  - Efficient way to represent **linguistic** and **subjective** attributes of the real world in computing
  - Supports smooth transitions between behaviors
  - Generally easier to create than methods involving training
- **Cons**
  - Defining set membership functions can be difficult
  - Debugging knowledge can be difficult
  - Defuzzification step can have surprising subtleties