technische universität dortmund

Computational Intelligence

Winter Term 2011/12

Prof. Dr. Günter Rudolph

Lehrstuhl für Algorithm Engineering (LS 11)

Fakultät für Informatik

TU Dortmund

- Approximate Reasoning
- Fuzzy Control

So far:

- p: IF *X* is A THEN *Y* is B

  $\rightarrow$ R(x, y) = Imp(A(x), B(y)) — rule as relation; fuzzy implication

- rule: IF *X* is A THEN *Y* is B
  fact: *X* is A‘
  conclusion: *Y* is B‘

  $\rightarrow$ B‘(y) = $\sup_{x \in X}$ t(A‘(x), R(x, y)) — composition rule of inference

Thus:

- B‘(y) = $\sup_{x \in X}$ t(A‘(x), Imp(A(x), B(y)))

here:

$$A'(x) = \begin{cases} 1 & \text{for } x = x_0 \\ 0 & \text{otherwise} \end{cases}$$

crisp input!

$$B'(y) = \sup_{x \in X} t(\, A'(x), \mathrm{Imp}(\, A(x), B(y)\,)\,)$$

$$= \begin{cases} \sup_{x \neq x_0} t(\, 0, \mathrm{Imp}(\, A(x), B(y)\,)\,) & \text{for } x \neq x_0 \\ t(\, 1, \mathrm{Imp}(\, A(x_0), B(y)\,)\,) & \text{for } x = x_0 \end{cases}$$

$$= \begin{cases} 0 & \text{for } x \neq x_0 & \text{since } t(0, a) = 0 \\ \mathrm{Imp}(\,(\, A(x_0), B(y)\,) & \text{for } x = x_0 & \text{since } t(a, 1) = a \end{cases}$$

Lemma:

a) $t(a, 1) = a$

b) $t(a, b) \le \min\{ a, b \}$

c) $t(0, a) = 0$

Proof:

by a)

ad a) Identical to axiom 1 of t-norms.

ad b) From monotonicity (axiom 2) follows for $b \le 1$, that $t(a, b) \le t(a, 1) = a$. Commutativity (axiom 3) and monotonicity lead in case of $a \le 1$ to $t(a, b) = t(b, a) \le t(b, 1) = b$. Thus, $t(a, b)$ is less than or equal to a as well as b, which in turn implies $t(a, b) \le \min\{ a, b \}$.

ad c) From b) follows $0 \le t(0, a) \le \min\{ 0, a \} = 0$ and therefore $t(0, a) = 0$. ■

Multiple rules:

IF X is A_1, THEN Y is B_1	$\to R_1(x, y) = Imp_1(A_1(x), B_1(y))$
IF X is A_2, THEN Y is B_2	$\to R_2(x, y) = Imp_2(A_2(x), B_2(y))$
IF X is A_3, THEN Y is B_3	$\to R_3(x, y) = Imp_3(A_3(x), B_3(y))$
…	…
IF X is A_n, THEN Y is B_n	$\to R_n(x, y) = Imp_n(A_n(x), B_n(y))$
X is A‘	

Y is B‘

Multiple rules for <u>crisp input</u>: x_0 is given

$B_1‘(y) = Imp_1(A_1(x_0), B_1(y))$
…
$B_n‘(y) = Imp_n(A_n(x_0), B_n(y))$

aggregation of rules or local inferences necessary!

aggregate! $\Rightarrow$ $B‘(y) = \textbf{aggr}\{ B_1‘(y), \ldots, B_n‘(y) \}$, where $\textbf{aggr} = \begin{cases} \min \\ \max \end{cases}$

<u>**FITA**: “**F**irst **i**nference, **t**hen **a**ggregate!“</u>

1. Each rule of the form IF *X* is A_k THEN *Y* is B_k must be transformed by an appropriate fuzzy implication $Imp_k(\cdot, \cdot)$ to a relation R_k : $R_k(x, y) = Imp_k(A_k(x), B_k(y))$.
2. Determine $B_k‘(y) = R_k(x, y) \circ A‘(x)$ for all $k = 1, \ldots, n$ (locale inference).
3. Aggregate to $B‘(y) = \beta(B_1‘(y), \ldots, B_n‘(y))$.

<u>**FATI**: “**F**irst **a**ggregate, **t**hen **i**nference!“</u>

1. Each rule of the form IF *X* ist A_k THEN *Y* ist B_k must be transformed by an appropriate fuzzy implication $Imp_k(\cdot, \cdot)$ to a relation R_k : $R_k(x, y) = Imp_k(A_k(x), B_k(y))$.
2. Aggregate $R_1, \ldots, R_n$ to a **superrelation** with aggregating function $\alpha(\cdot)$**:** $R(x, y) = \alpha(R_1(x, y), \ldots, R_n(x, y))$.
3. Determine $B‘(y) = R(x, y) \circ A‘(x)$ w.r.t. superrelation (inference).

1. Which principle is better? FITA or FATI?

2. Equivalence of FITA and FATI ?

FITA:

$$B‘(y) = \beta(B_1‘(y), \ldots, B_n‘(y)) = \beta(R_1(x, y) \circ A‘(x), \ldots, R_n(x, y) \circ A‘(x))$$

FATI:

$$B‘(y) = R(x, y) \circ A‘(x) = \alpha(R_1(x, y), \ldots, R_n(x, y)) \circ A‘(x)$$

special case:

$$A'(x) = \begin{cases} 1 & \text{for } x = x_0 \\ 0 & \text{otherwise} \end{cases}$$

crisp input!

On the equivalence of FITA and FATI:

FITA: $B'(y) = \beta(\, B_1'(y), \ldots, B_n'(y)\,)$

$= \beta(\, Imp_1(A_1(x_0), B_1(y)\,), \ldots, Imp_n(A_n(x_0), B_n(y)\,)\,)$

FATI: $B'(y) = R(x, y) \circ A'(x)$

$= \sup_{x \in X} t(\, A'(x), R(x, y)\,)$ (from now: special case)

$= R(x_0, y)$

$= \alpha(\, Imp_1(\, A_1(x_0), B_1(y)\,), \ldots, Imp_n(\, A_n(x_0), B_n(y)\,)\,)$

evidently: sup-t-composition with arbitrary t-norm and $\alpha(\cdot) = \beta(\cdot)$

- **AND-connected premises**

IF $X_1 = A_{11}$ AND $X_2 = A_{12}$ AND … AND $X_m = A_{1m}$ THEN $Y = B_1$
…
IF $X_n = A_{n1}$ AND $X_2 = A_{n2}$ AND … AND $X_m = A_{nm}$ THEN $Y = B_n$

reduce to single premise for each rule k:

$A_k(x_1,\ldots, x_m) = \min \{ A_{k1}(x_1), A_{k2}(x_2), \ldots, A_{km}(x_m) \}$ or in general: t-norm

- **OR-connected premises**

IF $X_1 = A_{11}$ OR $X_2 = A_{12}$ OR … OR $X_m = A_{1m}$ THEN $Y = B_1$
…
IF $X_n = A_{n1}$ OR $X_2 = A_{n2}$ OR … OR $X_m = A_{nm}$ THEN $Y = B_n$

reduce to single premise for each rule k:

$A_k(x_1,\ldots, x_m) = \max \{ A_{k1}(x_1), A_{k2}(x_2), \ldots, A_{km}(x_m) \}$ or in general: s-norm

important:

- if rules of the form **IF *X* is A THEN *Y* is B** interpreted as <u>logical</u> implication

⇒ $R(x, y) = Imp(\, A(x), B(y)\,)$ makes sense

- we obtain: $B'(y) = \sup_{x \in X} t(\, A'(x), R(x, y)\,)$

⇒ the worse the match of premise A'(x), the larger is the fuzzy set B'(y)

⇒ follows immediately from axiom 1: $a \le b$ implies $Imp(a, z) \ge Imp(b, z)$

interpretation of output set B'(y):

- B'(y) is the set of values that are still possible
- each rule leads to an additional restriction of the values that are still possible

⇒ resulting fuzzy sets $B'_k(y)$ obtained from single rules must be mutually <u>intersected</u>!

⇒ aggregation via $B'(y) = \mathbf{min} \{ B_1'(y), \ldots, B_n'(y) \}$

important:

- if rules of the form **IF *X* is A THEN *Y* is B** are <u>not</u> interpreted as <u>logical</u> implications, then the function Fct(·) in

$$R(x, y) = Fct(\, A(x), B(y)\,)$$

can be chosen as required for desired interpretation.

- frequent choice (especially in fuzzy control):
  - $R(x, y) = \min \{ A(x), B(x) \}$ Mamdami – "implication"
  - $R(x, y) = A(x) \cdot B(x)$ Larsen – "implication"

⇒ of course, they are no implications but special t-norms!

⇒ thus, if <u>relation R(x, y) is given</u>,
then the *composition rule of inference*

$$B'(y) = A'(x) \circ R(x, y) = \sup_{x \in X} \min \{ A'(x), R(x, y) \}$$

still can lead to a conclusion via fuzzy logic.

example: [JM96, S. 244ff.]

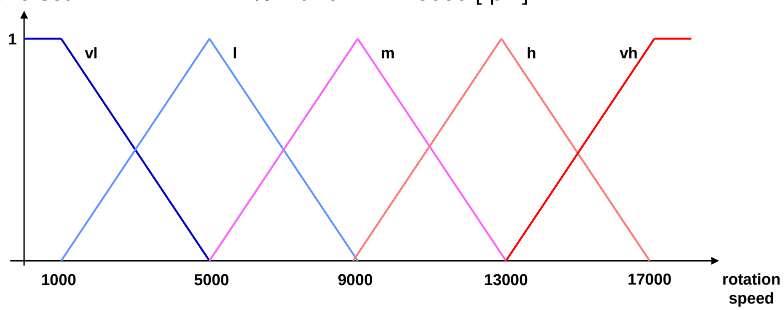
industrial drill machine → control of cooling supply

modelling

linguistic variable : **rotation speed**

linguistic terms : ***very low***, ***low***, ***medium***, ***high***, ***very high***

ground set : $\mathcal{X}$ with $0 \le x \le 18000$ [rpm]



example: (continued)

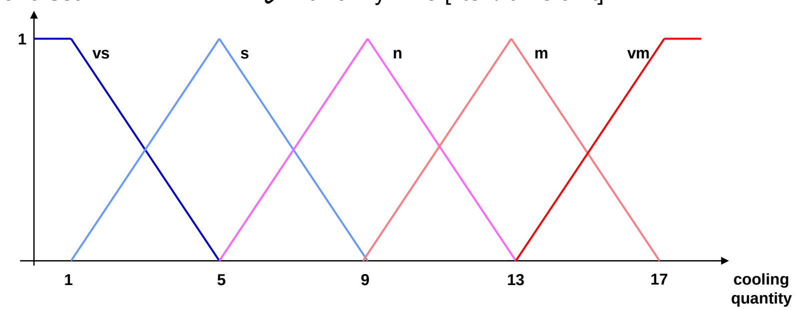
industrial drill machine → control of cooling supply

modelling

linguistic variable : **cooling quantity**

linguistic terms : ***very small***, ***small***, ***normal***, ***much***, ***very much***

ground set : $\mathcal{Y}$ with $0 \le y \le 18$ [liter / time unit]



example: (continued)

industrial drill machine → control of cooling supply

rule base

IF rotation speed IS	***very low***	**THEN cooling quantity IS**	***very small***
	low		***small***
	medium		***normal***
	high		***much***
	very high		***very much***

↑ sets $S_{vl}, S_l, S_m, S_h, S_{vh}$ "rotation speed"

↑ sets $C_{vs}, C_s, C_n, C_m, C_{vm}$ "cooling quantity"

example: (continued)

industrial drill machine → control of cooling supply

1. **input**: crisp value $x_0 = 10000$ min^{-1} (no fuzzy set!)
   → **fuzzyfication** = determine membership for each fuzzy set over $\mathcal{X}$
   → yields $S' = (0, 0, \frac{3}{4}, \frac{1}{4}, 0)$ via $x \mapsto (S_{vl}(x_0), S_l(x_0), S_m(x_0), S_h(x_0), S_{vh}(x_0))$
2. FITA: locale **inference** ⇒ since Imp(0,a) = 0 we only need to consider:
   S_m: $C'_n(y) = \text{Imp}(\frac{3}{4}, C_n(y))$
   S_h: $C'_m(y) = \text{Imp}(\frac{1}{4}, C_m(y))$
3. **aggregation**:
   $C'(y) = $ **aggr** $\{ C'_n(y), C'_m(y) \} = \max\{ \text{Imp}(\frac{3}{4}, C_n(y)), \text{Imp}(\frac{1}{4}, C_m(y)) \}$

? ?

example: (continued)

industrial drill machine → control of cooling supply

$C'(y) = \max \{ \mathrm{Imp}(\tfrac{3}{4}, C_n(y)), \mathrm{Imp}(\tfrac{1}{4}, C_m(y)) \}$

in case of <u>control task</u> typically no logic-based interpretation:

→ max-aggregation and

→ relation R(x,y) not interpreted as implication.

often: R(x,y) = min(a, b) „Mamdani controller“

thus:

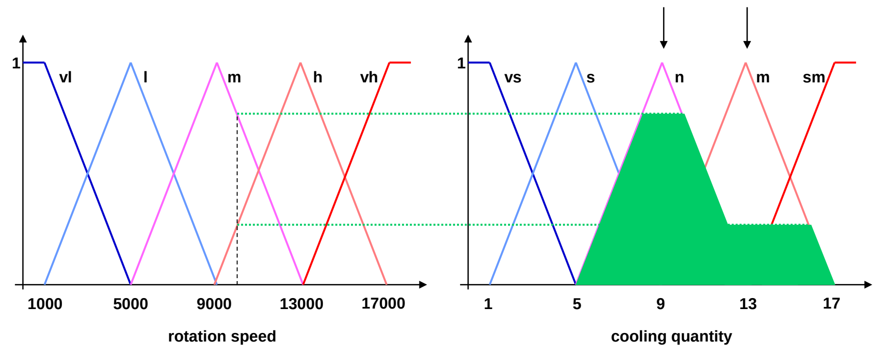
$C'(y) = \max \{ \min \{ \tfrac{3}{4}, C_n(y) \}, \min \{ \tfrac{1}{4}, C_m(y) \} \}$

→ graphical illustration

example: (continued)

industrial drill machine → control of cooling supply

$C'(y) = \max \{ \min \{ \tfrac{3}{4}, C_n(y) \}, \min \{ \tfrac{1}{4}, C_m(y) \} \}$, $x_0 = 10000$ [rpm]



open and closed loop control:

affect the dynamical behavior of a system
in a desired manner

- **open loop control**

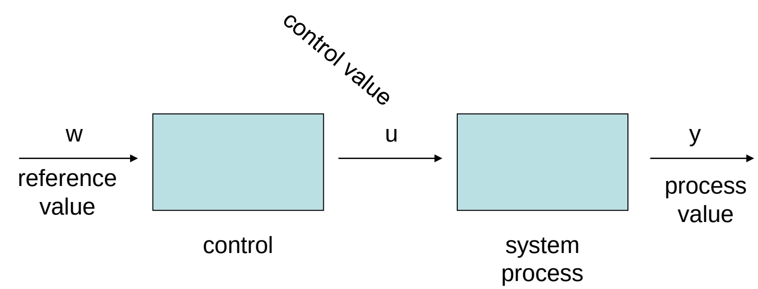
  control is aware of reference values and has a model of the system
  ⇒ control values can be adjusted,
  such that process value of system is equal to reference value

  problem: noise! ⇒ deviation from reference value not detected

- **closed loop control**

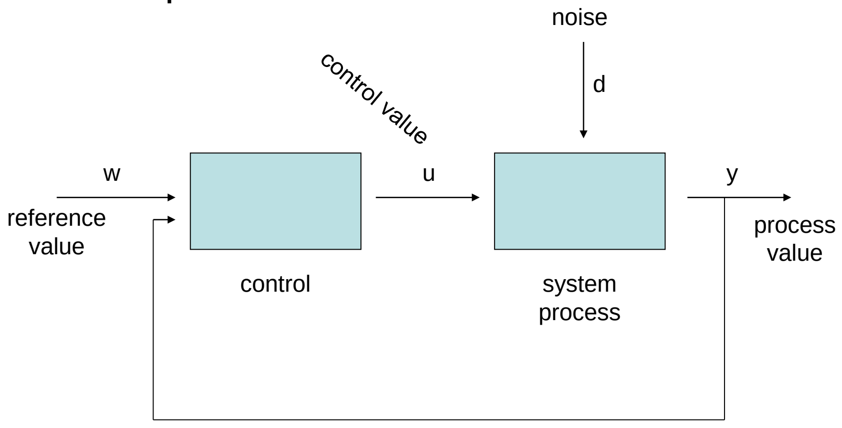
  now: detection of deviations from reference value possible
  (by means of measurements / sensors)
  and new control values can take into account the amount of deviation

open loop control



assumption: undisturbed operation ⇒ process value = reference value

closed loop control



control deviation = reference value – process value

required:

model of system / process

→ as differential equations or difference equations (DEs)

→ well developed theory available

so, why fuzzy control?

- there exists no process model in form of DEs etc. (operator/human being has realized control by hand)
- process with high-dimensional nonlinearities → no classic methods available
- control goals are vaguely formulated („soft“ changing gears in cars)

fuzzy description of control behavior

IF X is A_1, THEN Y is B_1
IF X is A_2, THEN Y is B_2
IF X is A_3, THEN Y is B_3
…
IF X is A_n, THEN Y is B_n
X is A‘

Y is B‘

} similar to approximative reasoning

but fact A‘ is not a fuzzy set but a crisp input

→ actually, it is the current process value

fuzzy controller executes inference step

→ yields fuzzy output set B‘(y)

but crisp control value required for the process / system

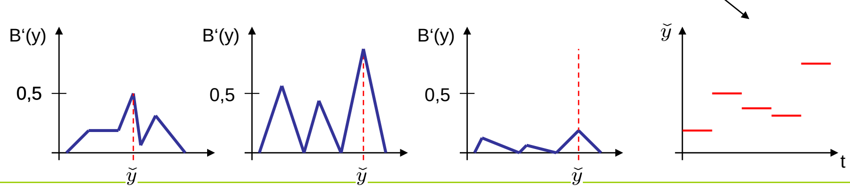
→ defuzzification (= “condense” fuzzy set to crisp value)

defuzzification

Def: rule k active ⇔ $A_k(x_0) > 0$

- maximum method
  - only active rule with largest activation level is taken into account
    - → suitable for pattern recognition / classification
    - → decision for a single alternative among finitely many alternatives
  - selection independent from activation level of rule (0.05 vs. 0.95)
  - if used for control: incontinuous curve of output values (leaps)

$$\breve{y} = \text{argmax } B'(y)$$

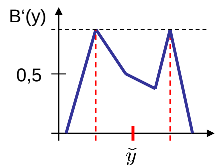


defuzzification

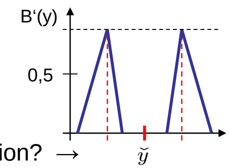
Y* = { y ∈ Y: B'(y) = hgt(B') }

- <u>maximum mean value method</u>
  - all active rules with largest activation level are taken into account
    - → interpolations possible, but need not be useful
    - → obviously, only useful for neighboring rules with max. activation
  - selection independent from activation level of rule (0.05 vs. 0.95)
  - if used in control: incontinuous curve of output values (leaps)

$$\breve{y} = \frac{1}{|Y^*|} \sum_{y^* \in Y^*} y^*$$



useful solution? →

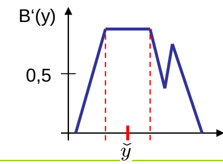


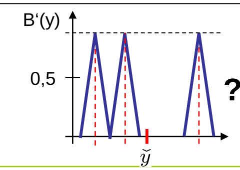
defuzzification

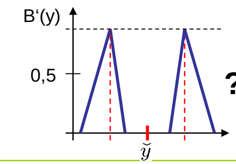
Y* = { y ∈ Y: B'(y) = hgt(B') }

- <u>center-of-maxima method (COM)</u>
  - only **extreme** active rules with largest activation level are taken into account
    - → interpolations possible, but need not be useful
    - → obviously, only useful for neighboring rules with max. activation level
  - selection indepependent from activation level of rule (0.05 vs. 0.95)
  - in case of control: incontinuous curve of output values (leaps)

$$\breve{y} = \frac{\inf Y^* + \sup Y^*}{2}$$







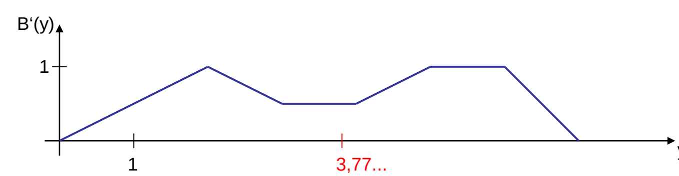
defuzzification

- Center of Gravity (COG)
  - all active rules are taken into account
    - → but numerically expensive … …only valid for HW solution, today!
    - → borders cannot appear in output (∃ work-around)
  - if only single active rule: independent from activation level
  - continuous curve for output values

$$\breve{y} = \frac{\int y \cdot B'(y)\, dy}{\int B'(y)\, dy}$$

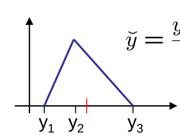
Excursion: COG

$$\breve{y} = \frac{\int y \cdot B'(y)\, dy}{\int B'(y)\, dy}$$



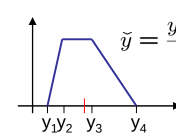
pendant in probability theory: expectation value

triangle:

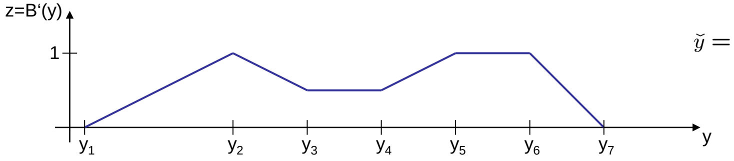


$$\breve{y} = \frac{y_1 + y_2 + y_3}{3}$$

trapezoid:



$$\breve{y} = \frac{y_4^2 + y_3^2 - y_2^2 - y_1^2 + y_3 y_4 - y_1 y_2}{3\,(y_4 + y_3 - y_2 - y_1)}$$



$$\breve{y} = \frac{\int y \cdot B'(y)\, dy}{\int B'(y)\, dy}$$

<u>assumption:</u> fuzzy membership functions piecewise linear

output set B'(y) represented by sequence of points $(y_1, z_1), (y_2, z_2), \ldots, (y_n, z_n)$

⇒ area under B'(y) and weighted area can be determined additively piece by piece

⇒ linear equation z = m y + b ⇒ insert (y_i, z_i) and (y_{i+1}, z_{i+1})

⇒ yields m and b for each of the n-1 linear sections

$$\Rightarrow F_i = \int_{y_i}^{y_{i+1}} (m\,y + b)\, dy = \frac{m}{2}(y_{i+1}^2 - y_i^2) + b(y_{i+1} - y_i)$$

$$\Rightarrow G_i = \int_{y_i}^{y_{i+1}} y\,(m\,y + b)\, dy = \frac{m}{3}(y_{i+1}^3 - y_i^3) + \frac{b}{2}(y_{i+1}^2 - y_i^2)$$

$$\breve{y} = \frac{\sum_i G_i}{\sum_i F_i}$$

Defuzzification

- Center of Area (COA)
  - developed as an approximation of COG
  - let $\hat{y}_k$ be the COGs of the output sets $B'_k(y)$:

$$\breve{y} = \frac{\sum_k A_k(x_0) \cdot \hat{y}_k}{\sum_k A_k(x_0)}$$