Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Vizualizace činnosti fuzzy regulátoru

2019
Vedoucí práce: doc. RNDr. Miroslav Kolařík, Ph.D.

Alena Svobodová
Studijní obor: Aplikovaná informatika, prezenční forma

Bibliografické údaje

Autor:	Alena Svobodová
Název práce:	Vizualizace činnosti fuzzy regulátoru
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2019
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	doc. RNDr. Miroslav Kolařík, Ph.D.
Počet stran:	41
Přílohy:	1 CD
Jazyk práce:	český

Bibliograhic info

Author:	Alena Svobodová
Title:	Visualisation of the way how fuzzy controllers work
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2019
Study field:	Applied Computer Science, full-time form
Supervisor:	doc. RNDr. Miroslav Kolařík, Ph.D.
Page count:	41
Supplements:	1 CD
Thesis language:	Czech

Anotace

Práce seznamuje čtenáře se základy fuzzy logiky, vysvětluje princip fungování fuzzy regulátoru a popsuje několik aplikací fuzzy logiky i regulátoru. Aplikace, která je součástí této bakalářské práce, vizualizuje vnitřní fungování regulátoru řídícího inverzní kyvadlo. Důraz je kladen na jednoduchost a výukovou stránku. Technologie použité pro realizaci jsou JavaFX, FXML a CSS.

Synopsis

The main objective of my thesis is to describe fuzzy logic, explain the inner workings (mechanism) of a fuzzy controller and outline a few out of many various applications of fuzzy logic and fuzzy controllers. The program, which is a part of this thesis, visualizes the way how a fuzzy controller used to control an inverted pendulum works. The technologies used to make this application are JavaFX, FXML and CSS.

Klíčová slova: fuzzy logika; fuzzy regulátor; inverzní kyvadlo; JavaFX

Keywords: fuzzy logic; fuzzy controller; inverted pendulum; JavaFX

Děkuji vedoucímu mé práce doc. RNDr. Miroslavu Kolaříkovi, Ph.D. za podnětné rady i připomínky. Dále bych chtěla poděkovat své rodině a přátelům za podporu při studiu.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

1 Úvod

Klasické techniky systémové analýzy jsou nevhodné pro práci s lidskými systémy, nebo systémy s podobnou mírou složitosti. Čím je vyšší složitost, tím je naše schopnost přesného vyjadřování a přesných úsudků o chování daného systému nižší. Proto zřejmě není přesná analýza použitelná v oblastech reálného světa, jako jsou například sociální, politické, ekonomické či jakékoliv problémy, kde hraje roli lidský faktor.

Alternativní přístup staví na myšlence, že důležité prvky v lidském uvažování nejsou čísla, ale označení fuzzy množin, tedy třídy objektů, u kterých je přechod mezi jednotlivými třídami plynulý, nikoliv náhlý. Toto naznačuje, že většina našeho uvažování nepodléhá zákonům klasické dvouhodnotové, či vícehodnotové logiky, ale logiky vágní, mlhavé – fuzzy. Podle Lotfi A. Zadeha, který je autorem konceptu fuzzy logiky, je právě ona vágní logika klíčová v nejdůležitější schopnosti lidského myšlení, ve schopnosti souhrnu informací. Tento přístup byl velmi zdárně uplatněn v mnoha odvětvích, z nichž nejúspěšnější aplikací je fuzzy řízení.

Cílem této bakalářské práce je seznámit čtenáře právě s tímto zajímavým alternativním přístupem, který nabízí mnoho výhod oproti klasickému řízení. Fuzzy regulátory jsou v dnešní době používané v řadě běžných řídících systémů od kuchyňských spotřebičů po auta, a mají obrovský potenciál pro budoucí využití.

Součástí práce je aplikace, která slouží pro vizualizaci vnitřního fungování regulátoru, který řídí inverzní kyvadlo. Aplikace je rozdělena podle fází výpočtu. Pomocí jednoduchého uživatelského rozhraní a minima ovládacích prvků si může uživatel vyzkoušet všechny fáze a průběh výpočtu. Aplikace byla naprogramována v jazyce Java kvůli přenositelnosti mezi operačními systémy. Při grafickém návrhu aplikace byl kladen důraz na jednoduchost a výukovou stránku.

V první části textu je úvod do fuzzy logiky, definice pojmů a operací. Další část textu je věnována fuzzy regulátoru a aplikacím. V následující části textu se nachází programátorská dokumentace, kde jsou popsané použité technologie a celková struktura aplikace. Nakonec je uvedena uživatelská dokumentace, která poskytuje základní informace k používání aplikace.

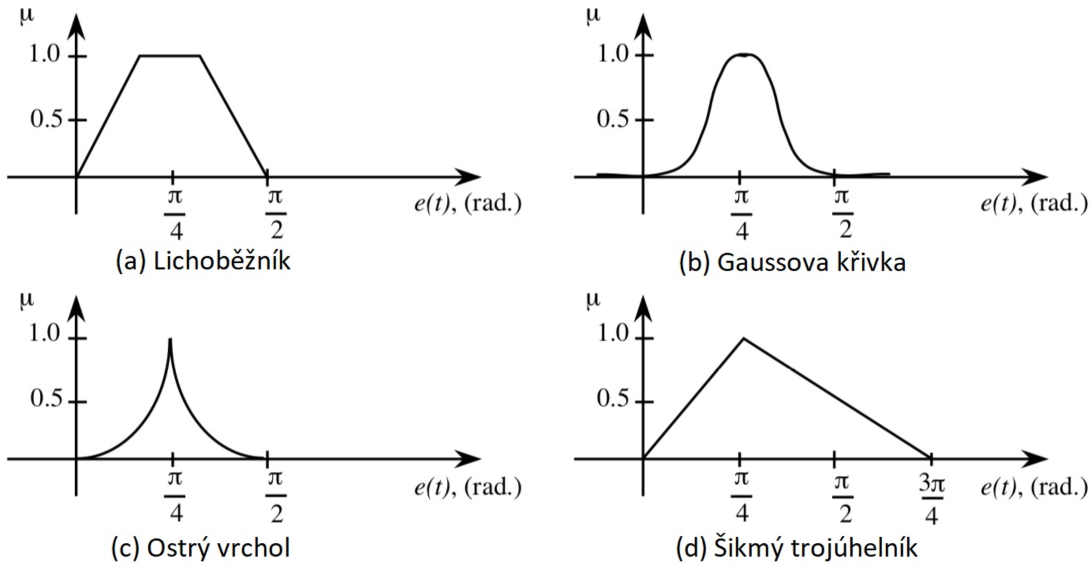
2 Fuzzy logika

2.1 Základy

Tradiční přístup k neurčitosti byl vždy takový, že neurčitost je nežádoucí a měli bychom se jí vyhýbat. Nepřesnost či nejasnost byla vnímána jako nevědecká. Tento pohled se začal v průběhu 19. století měnit. Vznikl nový alternativní pohled, který toleruje nepřesnost. Dokonce říká že je nevyhnutelnou součástí vědy, a může navíc být velmi užitečná. Začátkem 19. století byly ve fyzice zkoumány procesy na úrovni molekul. Do té doby byla používána přesná pravidla Newtonovské mechaniky, jejichž aplikace vyžadovala obrovské množství výpočtů, jež značně přesahovaly tehdejší možnosti. Bylo nutné najít koncepčně odlišný přístup umožňující zkoumání procesů právě na úrovni molekul. Následoval vývoj statistických metod, které nahrazují specifické projevy mikroskopických entit pomocí jejich statistických průměrů. Zejména teorie pravděpodobnosti nahradila přesné výpočty. Podstata teorie pravděpodobnosti je zachytit určitý typ nepřesností. Přesné výpočty vyžadovaly problémy s malým počtem proměnných, které spolu souvisí předvídatelně. Naopak teorie pravděpodobnosti potřebuje rozsáhlé množství dat a vysoký podíl náhody. Tyto dva přístupy se doplňují, avšak dokáží pokrýt jen dva extrémy v ohledu komplexnosti a náhodnosti problémů. Warren Weaver je označuje jako organizovaná jednoduchost a neorganizovaná složitost [1]. V praxi se však nejčastěji setkáváme s problémy typu organizovaná složitost. Vývoj počítačových technologií v období druhé světové války umožnil řešit čím dál složitější problémy, jež připomínaly typ organizovaná složitost. Věřilo se, že míra složitosti problémů, které je možné řešit, je jen otázkou výpočetní síly, jež je k dispozici. Později, v 60. letech, se ukázalo, že existují hranice toho, jak složité problémy se dají vypočítat s využitím počítačových technologií. Bremermannova mez je jedna z takových hranic. Číslo 10^{93} je počet bitů, které by byl maximálně schopen zpracovat počítač o velikosti planety Země a výpočtem délky stáří planety Země [2]. Nabízí se tedy otázka, jak řešit problémy, na jejichž složitost nemáme dostatek výpočetní síly. Při tvorbě modelů skutečnosti často využíváme nepřesnost, která nepřímo souvisí se složitostí modelu a přímo souvisí s důvěryhodností modelu. V 60. letech tak vzrostlo uznání důležité role neurčitosti, jež odstartovalo druhou vlnu přechodu od tradičního k modernímu pohledu na neurčitost. V roce 1965 publikoval Lotfi A. Zadeh článek, který představoval teorii, jejíž objekty, *fuzzy množiny*, nemají na rozdíl od klasických množin přesné hranice. Tedy příslušnost ve fuzzy množině není dána dvouhodnotově 1 – pravda, 0 – nepravda, nýbrž stupněm příslušnosti. Obrovský význam Zadehova článku spočívá v tom, že zpochybnil nejen teorii pravděpodobnosti, ale i základní myšlenky, na kterých je postavena, na Aristotelovské dvouhodnotové logice [3].

2.2 Fuzzy množina

Fuzzy množina je třída objektů bez přesně definovaného kritéria příslušnosti. Je charakterizována funkcí příslušnosti, která přiřazuje každému objektu stupeň



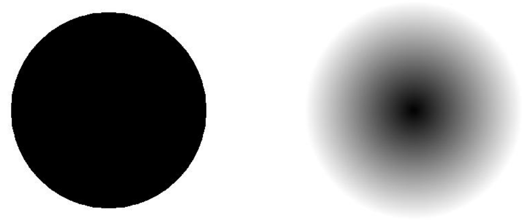
Obrázek 1: Vybrané typy funkcí příslušnosti [5]

příslušnosti z uzavřeného intervalu 0 až 1. Relace inkluze, operace sjednocení, průnik a komplement jsou generalizované na tyto množiny. Velmi často se v reálném světě stává, že objekty nemají přesně definována kritéria příslušnosti. Například třída zvířat počítá psy, koně, ptáky jako její členy, ale kameny, tekutiny nebo rostliny ne. Naproti tomu u mořské hvězdice či bakterie není zřejmé, zda do třídy zvířat patří. Podobně není zřejmé, zda číslo 10 patří do třídy všech reálných čísel mnohem větších než 1. Další příklady jsou třída „vysokých mužů“, třída „vysoké teploty“, nebo třída „červených objektů“ [4].

Funkce příslušnosti přiřazuje fuzzy množinám hodnoty z uzavřeného intervalu, nejčastěji 0 až 1 [1]. Hodnoty pak označují stupeň příslušnosti prvků. Vyšší hodnoty znamenají vyšší stupeň příslušnosti k dané množině. Funkce příslušnosti jsou jedinečné pro každý případ a záleží na kontextu. Například koncept „vysoká teplota“ představuje odlišné hodnoty v kontextu počasí a v kontextu teplot jaderného reaktoru. Existují různé typy funkcí příslušnosti (viz obrázek 1). V praxi se ukazuje, že nejsou velké rozdíly při použití těchto různých typů, a tak se upřednostňují jednodušší varianty [3].

Význam fuzzy množin spočívá v jejich možnostech vyjádřit plynulý přechod mezi tím, kdy objekt patří a kdy nepatří do dané množiny. Princip fuzzy množin je velmi blízký lidskému vyjadřování. Například místo popisování oblačnosti v přesných procentech říkáme, že je „slunečno“. Přesto, že takové vyjádření je vágní a nepřesné, je v mnoha případech vhodnější. Klasické množiny nám umožňují vyjádřit ostré hranice mezi skupinami. V přirozeném jazyce jsou však běžná označení jako „vysocí lidé“, „drahá auta“, „vysoce nakažlivá nemoc“, „skromné

[1]Mezi další používané intervaly patří například 0 až 255, avšak interval 0 až 1 je nejpoužívanější a tak se text zabývá pouze tímto.



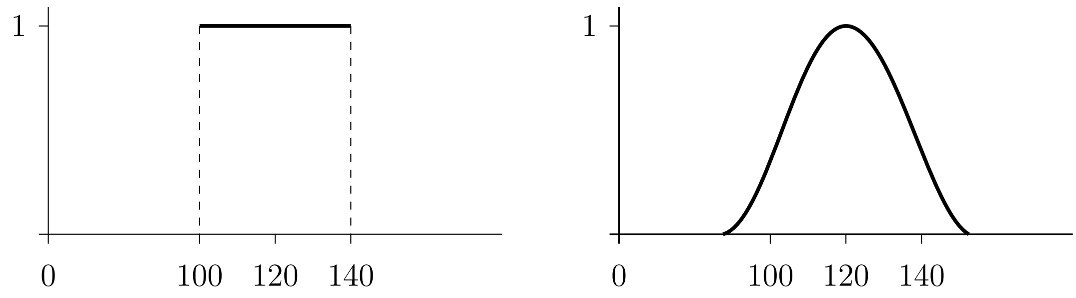
Obrázek 2: Klasiká množina a fuzzy množina

zisky“ a „slunečné dny“. U skupin s takovým označením není přesně daná hranice, kdy objekt náleží k dané třídě. Fuzzy množina nám dává možnost přiřadit objektu míru udávající stupeň kompatibility s danou třídou. Tato míra koresponduje se stupněm podobnosti, neboli kompatibility s konceptem reprezentovaným danou fuzzy množinou. Tedy fuzzy množina reprezentující „slunečno“ by mohla přiřadit stupeň 1 k 0procentní oblačnosti, stupeň 0,8 k 20procentní oblačnosti, stupeň 0,4 ke 30procentní oblačnosti a stupeň 0 k oblačnosti 75procentní. Takové míry reprezentují naše subjektivní vnímání konceptu „slunečno“ [3].

Fuzzy logika dokáže popsat mnoho problémů, které v klasické dvouhodnotové logice vedou k paradoxům. Příkladem je paradox hromady. Jedno zrnko písku není hromada a když přidáme druhé zrnko, tak hromada nevznikne. Z toho logicky vyvodíme, že opakovaným přidáním jednoho zrnka nikdy nevznikne hromada, což není pravda. Obdobně je tomu v opačném případě, když máme hromadu a odebereme jedno zrnko, i potom je to stále hromada. Provedeme mnoho opakování tohoto postupu a nakonec nezbyde nic. V jaký okamžik to přestala být hromada? Klasická dvouhodnotová logika nestačí na zachycení vlastnosti *být hromadou*, nestačí vyjádření „je pravda“ nebo „není pravda“ [6].

„Fuzzy značí v angličtině cosi nejasného, neostrého, mlhavého. Po řadě pokusů o výstižný překlad se ve většině jazyků ujal termín fuzzy, u nejž aspoň nehrozí jiná interpretace“ [6]. Pro zajímavost uvádíme, že v angličtině působí slovo fuzzy lehce negativně, a tak se pro výrobky přednostně nepoužívá.

Je důležité upozornit na dva druhy neurčitosti, které je třeba od sebe rozlišovat, a to *pravděpodobnost* a *vágnost informace*. Nejlépe lze rozdíl ilustrovat na příkladu: Jdeme do restaurace na oběd. Pravděpodobnost, že oběd dostaneme může ovlivnit řada faktorů. Např. restaurace může být zavřená nebo zcela obsazená. V tomto případě hovoříme o pravděpodobnosti. Nyní provedeme pokus zajdeme do restaurace a dostaneme jednoznačnou odpověď na naši otázku. Tady již není neurčitost, již víme přesně, jestli jsme oběd dostali, nebo ne. Naproti tomu, když se zeptáme, jestli byl oběd dobrý, dostaneme různé odpovědi od různých lidí na celé stupnici hodnot. K jejich zachycení jistě nestačí klasická dvouhodnotová logika. Je nutné dodat, že ani žádným dalším pokusem odpověď



Obrázek 3: Grafy klasické množiny a fuzzy množiny reprezentující *normální krevní tlak* [7]

nezpřesníme a neurčitosti se tak nezbavíme [6].

Klasická množina vs. fuzzy množina
Klasickou množinu A lze vyjádřit jako zobrazení

$$A : U \to \{0, 1\} ,$$

které každému prvku u z daného univerza U přiřadí 1, pokud u patří do A, a které přiřadí 0, pokud u nepatří do A. Fuzzy množinu A lze podobně vyjádřit jako zobrazení

$$A : U \to [0, 1] ,$$

které každému prvku u z U přiřadí stupeň $A(u)$ náležení prvku u do fuzzy množiny A. Viz obrázek 2 [7].

Příklad: Obrázek 3 ukazuje klasickou množinu a fuzzy množinu, které reprezentují pojem „normální krevní tlak“. Fuzzy množiny nemají ostré hranice mezi tím, kdy prvek přísluší k množině a kdy ne.

Další příklad fuzzy množiny. Nechť $U = \mathbb{R}, L = [0, 1]$. Uvažujme fuzzy množinu $A : U \to L$ definovanou takto:

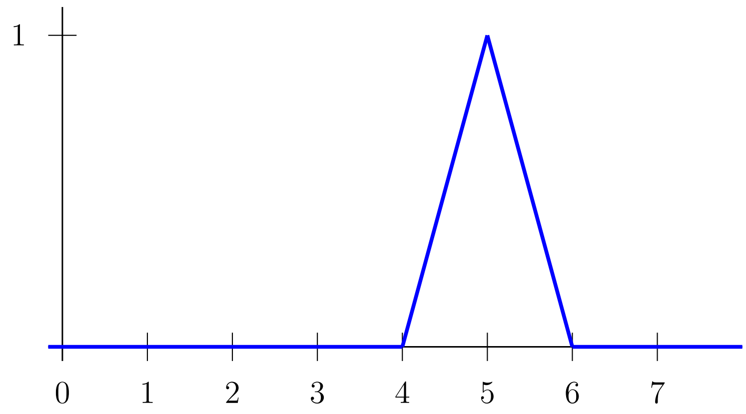
$$A(u) = \begin{cases} 0 & \text{pro } u < 4, \\ x - 4 & \text{pro } 4 \leq u \leq 5, \\ 6 - x & \text{pro } 5 < u \leq 6, \\ 0 & \text{pro } u > 6. \end{cases}$$

Tato fuzzy množina reprezentuje možný význam výrazu „přibližně 5“, viz obrázek 4 [7].

2.3 Operace s fuzzy množinami

V klasiké logice vyjadřují spojky logické funkce a lze je znázornit tabulkou viz tabulka 1.

Ve fuzzy logice musí konjunkce také přiradit dvěma pravdivostním hodnotám výslednou pravdivostní hodnotu.



Obrázek 4: Graf fuzzy množiny reprezentující *přibližně 5* [7]

p	q	negace: $\neg p$	konjunkce: $p \wedge q$	disjunkce: $p \vee q$	implikace: $p \Rightarrow q$	ekvivalence: $p \Leftrightarrow q$
1	1	0	1	1	1	1
1	0	0	0	1	0	0
0	1	1	0	1	1	0
0	0	1	0	0	1	1

Tabulka 1: Logické funkce

2.4 Základní pojmy

Základní pojmy pro libovolnou fuzzy množinu A na univerzu X:

- *Obor pravdivostních hodnot*, anglicky *range, level set*, značíme:

$$Range\,(A) = \{\alpha \in \langle 0,1 \rangle : (\exists x \in X : \mu_A\,(x) = \alpha)\}\,.$$

- *Výška*: $h(A) = supRange(A)$. Je-li fuzzy množina výšky 1 pak se nazývá *normální*, v opačném případě *subnormální*.

- *Nosič*, anglicky *support*, je ostrá[2] množina

$$Supp(A) = \{x \in X : \mu_A(x) > 0\},\ \text{neboli}\ Supp(A) = \mu_A^{-1}\left(\langle 0,1 \rangle\right).$$

- *Jádro*, anglicky *core* je ostrá množina

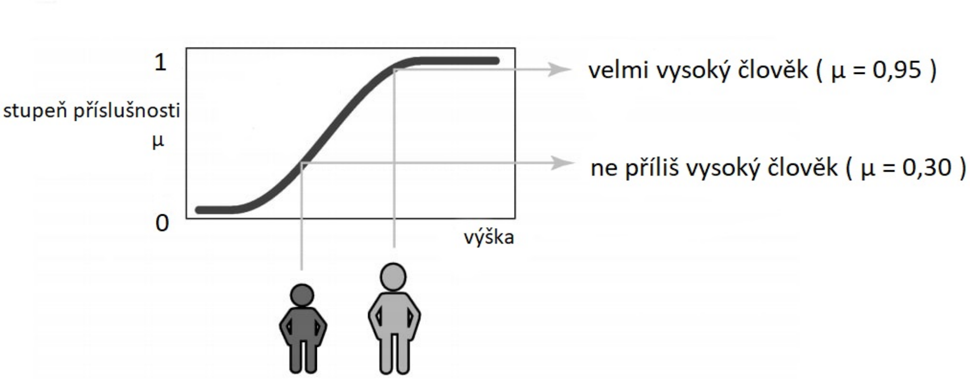
$$core(A) = \{x \in X : \mu_A(x) = 1\}\,,\ \text{tj.}\ core(A) = \mu_A^{-1}(1).$$

2.5 Operace

Základ pro operace s fuzzy množinami jsou operace fuzzy výrokového počtu, operace s pravdivostními hodnotami z intervalu $\langle 0,1 \rangle$.[3]

[2] Klasické množiny pro odlišení od fuzzy množin zde označujeme jako ostré.

[3] Operace s fuzzy množinami jsou zobecněním operací v klasické dvouhodnotové logice, takže pro ně používáme totéž značení.



Obrázek 5: Spojitá funkce znázorňující *být vysoký*

2.5.1 Fuzzy negace

Funkce příslušnosti k fuzzy množině říká, nakolik má prvek univerza určitou vlastnost. Negace říká, do jaké míry prvek tuto vlastnost nemá, ale nakolik má vlastnost opačnou.

Dá se dokázat, že fuzzy negace je unární operace $\underset{\cdot}{\neg} : \langle 0,1\rangle \to \langle 0,1\rangle$, splňující následující axiomy:

$$\alpha \leq \beta \quad \Rightarrow \quad \underset{\cdot}{\neg}\beta \leq \underset{\cdot}{\neg}\alpha, \tag{1}$$

$$\underset{\cdot}{\neg}\underset{\cdot}{\neg}\alpha \quad = \quad \alpha. \tag{2}$$

Fuzzy negace je antitonní podle axiomu (1) a involutivní podle axiomu (2). Standardní negace $\underset{s}{\neg}$ je definována vztahem $\underset{s}{\neg}\alpha = 1 - \alpha$. Z axiomů (1) a (2) vyplývá, že každá fuzzy negace je spojitá, klesající, bijektivní a splňuje okrajové podmínky

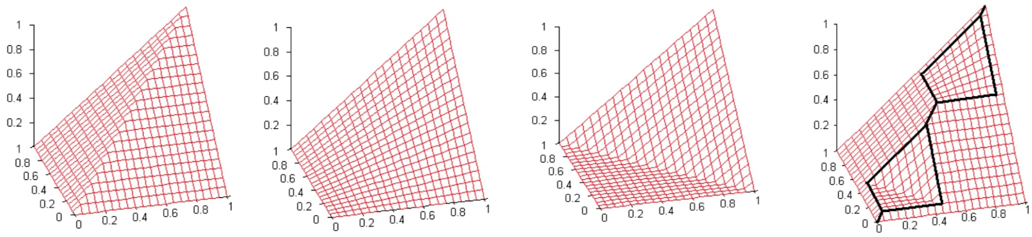
$$\underset{\cdot}{\neg}1 = 0, \quad \underset{\cdot}{\neg}0 = 1.$$

Její graf je symetrický podle osy 1. a 3. kvadrantu. Neboli $\underset{\cdot}{\neg}^{-1} = \underset{\cdot}{\neg}$, což znamená, že je funkce sama k sobě inverzní. Důkaz lze najít v textu [6].
Důsledek: Pro každou fuzzy negaci $\underset{\cdot}{\neg}$ existuje právě jedna hodnota $e \in (0,1)$ pro kterou $\underset{\cdot}{\neg}e = e$. Nazýváme ji rovnovážnou hodnotou (anglicky equilibrium).

2.5.2 Fuzzy konjunkce (trojúhelníkové normy)

Když nás zajímá do jaké míry je například výrobek dobrý a levný, tak nás zajímá fuzzy konjunkce dvou množin.

Dá se dokázat, že fuzzy konjunkce (trojúhelníková norma, t-norma, anglicky triangular norm) je binární operace $\underset{\cdot}{\wedge} : \langle 0,1\rangle^2 \to \langle 0,1\rangle$ splňující následující axiomy pro všechna $\alpha, \beta, \gamma \in \langle 0,1\rangle$:



Obrázek 6: Různé typy fuzzy konjunkcí

$$\alpha \underset{\cdot}{\wedge} \beta = \beta \underset{\cdot}{\wedge} \alpha \qquad \text{(komutativita)} \tag{3}$$

$$\alpha \underset{\cdot}{\wedge} (\beta \underset{\cdot}{\wedge} \gamma) = (\alpha \underset{\cdot}{\wedge} \beta) \underset{\cdot}{\wedge} \gamma \qquad \text{(asociativita)} \tag{4}$$

$$\beta \leq \gamma \Rightarrow \alpha \underset{\cdot}{\wedge} \beta \leq \alpha \underset{\cdot}{\wedge} \gamma \qquad \text{(monotonie)} \tag{5}$$

$$\alpha \underset{\cdot}{\wedge} 1 = \alpha \qquad \text{(okrajová podmínka)} \tag{6}$$

Příklady konjunkcí

- *Standardní* konjunkce (min, Zadehova):

$$\alpha \underset{S}{\wedge} \beta = \min(\alpha, \beta).$$

- *Łukasiewiczova* konjunkce (Gilesova, bold):

$$\alpha \underset{L}{\wedge} \beta = \begin{cases} \alpha + \beta - 1 & \text{pro } \alpha + \beta > 0, \\ 0 & \text{jinak.} \end{cases}$$

- *Součinová* konjunkce (produktová, pravděpodobnostní, algebraic):

$$\alpha \underset{P}{\wedge} \beta = \alpha \cdot \beta.$$

- *Drastická* konjunkce (slabá, weak):

$$\alpha \underset{D}{\wedge} \beta = \begin{cases} \alpha & \text{pro } \beta = 1, \\ \beta & \text{pro } \alpha = 1, \\ 0 & \text{jinak.} \end{cases}$$

Fuzzy konjunkcí je mnoho různých druhů a volba záleží na konkrétní situaci, která se nám bude nejvíce hodit pro co nejpřesnější vyjádření [6].

2.5.3 Fuzzy disjunkce (trojúhelníkové konormy)

Obdobně jako fuzzy konjunkce, fuzzy disjunkce je zobecněním klasické disjunkce, kterou známe z klasické logiky, na interval $\langle 0,1\rangle$.

Dá se dokázat, že fuzzy disjunkce (trojúhelníková konorma, t-konorma, anglicky triangular conorm) je binární operace $\dot\vee : \langle 0,1\rangle^2 \to \langle 0,1\rangle$ splňující následující axiomy pro všechna $\alpha, \beta, \gamma \in \langle 0,1\rangle$:

$$\alpha \mathbin{\dot\vee} \beta = \beta \mathbin{\dot\vee} \alpha \qquad \text{(komutativita)} \tag{7}$$

$$\alpha \mathbin{\dot\vee} (\beta \mathbin{\dot\vee} \gamma) = (\alpha \mathbin{\dot\vee} \beta) \mathbin{\dot\vee} \gamma \qquad \text{(asociativita)} \tag{8}$$

$$\beta \leq \gamma \Rightarrow \alpha \mathbin{\dot\vee} \beta \leq \alpha \mathbin{\dot\vee} \gamma \qquad \text{(monotonie)} \tag{9}$$

$$\alpha \mathbin{\dot\vee} 1 = \alpha \qquad \text{(okrajová podmínka)} \tag{10}$$

Příklady fuzzy disjunkcí:

- *Standardní* disjunkce (max, Zadehova):

$$\alpha \overset{S}{\vee} \beta = \max(\alpha, \beta).$$

- *Łukasiewiczova* disjunkce (Gilesova, bold, bounded sum):

$$\alpha \overset{L}{\vee} \beta = \begin{cases} \alpha + \beta & \text{pro } \alpha + \beta < 1, \\ 1 & \text{jinak.} \end{cases}$$

- *Součinová* disjunkce (produktová, pravděpodobnostní):

$$\alpha \overset{P}{\vee} \beta = \alpha + \beta - \alpha \cdot \beta.$$

- *Drastická* disjunkce (slabá, weak):

$$\alpha \overset{D}{\vee} \beta = \begin{cases} \alpha & \text{pro } \beta = 0, \\ \beta & \text{pro } \alpha = 0, \\ 1 & \text{jinak.} \end{cases}$$

Poznámka: Zadeh v textu [4] bral v úvahu pouze standardní operace, ale jiní autoři se fuzzy operacemi zabývali ještě dříve než se objevil pojem fuzzy množiny, protože jsou velmi intuitivní a dají se využít v mnoha jiných oblastech [6].

2.5.4 Logické spojky

Ze základních logických spojek lze odvodit další a tvořit logické formule. Níže je tabulka zákonů známá z klasické logiky upravená pro fuzzy logiku.

$$\begin{array}{lcc}
\text{involuce:} & \underset{\cdot}{\neg}(\underset{\cdot}{\neg}\alpha) = \alpha, & \\
\text{komutativita:} & \alpha \mathbin{\dot{\vee}} \beta = \beta \mathbin{\dot{\vee}} \alpha, & \\
 & \alpha \mathbin{\underset{\cdot}{\wedge}} \beta = \beta \mathbin{\underset{\cdot}{\wedge}} \alpha, & \\
\text{asociativita:} & (\alpha \mathbin{\dot{\vee}} \beta) \mathbin{\dot{\vee}} \gamma = \alpha \mathbin{\dot{\vee}} (\beta \mathbin{\dot{\vee}} \gamma), & \\
 & (\alpha \mathbin{\underset{\cdot}{\wedge}} \beta) \mathbin{\underset{\cdot}{\wedge}} \gamma = \alpha \mathbin{\underset{\cdot}{\wedge}} (\beta \mathbin{\underset{\cdot}{\wedge}} \gamma), & \\
\text{distributivita:} & \alpha \mathbin{\underset{\cdot}{\wedge}} (\beta \mathbin{\dot{\vee}} \gamma) = (\alpha \mathbin{\underset{\cdot}{\wedge}} \beta) \mathbin{\dot{\vee}} (\alpha \mathbin{\underset{\cdot}{\wedge}} \gamma), & \\
 & \alpha \mathbin{\dot{\vee}} (\beta \mathbin{\underset{\cdot}{\wedge}} \gamma) = (\alpha \mathbin{\dot{\vee}} \beta) \mathbin{\underset{\cdot}{\wedge}} (\alpha \mathbin{\dot{\vee}} \gamma), & \\
\text{idempotence:} & \alpha \mathbin{\dot{\vee}} \alpha = \alpha, & \alpha \mathbin{\underset{\cdot}{\wedge}} \alpha = \alpha, \\
\text{absorpce:} & \alpha \mathbin{\dot{\vee}} (\alpha \mathbin{\underset{\cdot}{\wedge}} \beta) = \alpha, & \\
 & \alpha \mathbin{\underset{\cdot}{\wedge}} (\alpha \mathbin{\dot{\vee}} \beta) = \alpha, & \\
\text{absorpce s jednotkou a nulou:} & \alpha \mathbin{\dot{\vee}} 1 = 1, & \alpha \mathbin{\underset{\cdot}{\wedge}} 0 = 0, \\
\text{neutrální prvky:} & \alpha \mathbin{\dot{\vee}} 0 = \alpha, & \alpha \mathbin{\underset{\cdot}{\wedge}} 1 = \alpha, \\
\text{zákon kontradikce:} & & \alpha \mathbin{\underset{\cdot}{\wedge}} \underset{\cdot}{\neg}\alpha = 0, \\
\text{zákon vyloučeného třetího:} & \alpha \mathbin{\dot{\vee}} \underset{\cdot}{\neg}\alpha = 1, & \\
\text{de Morganovy zákony:} & \underset{\cdot}{\neg}(\alpha \mathbin{\dot{\vee}} \beta) = \underset{\cdot}{\neg}\alpha \mathbin{\dot{\vee}} \underset{\cdot}{\neg}\beta, & \\
 & \underset{\cdot}{\neg}(\alpha \mathbin{\underset{\cdot}{\wedge}} \beta) = \underset{\cdot}{\neg}\alpha \mathbin{\dot{\wedge}} \underset{\cdot}{\neg}\beta. &
\end{array}$$

Protože existuje několik typů fuzzy operací je nutné upřesnit, které z nich splňují jednotlivé zákony. Všechny fuzzy negace, konjunkce a disjunkce splňují involuci, komutativitu, asociativitu, absorpci s jednotkou a nulou a mají neutrální prvky. Standardní negace, konjunkce a disjunkce splňují všechny kromě kontradikce a zákona vyloučeného třetího. Struktura se standardními operacemi se nazývá de Morganova algebra. Podrobněji o konkrétních operacích a zákonech, které pro ně platí se lze dočíst v textu [6].

2.5.5 Fuzzy implikace

I zde se můžeme setkat s různými typy fuzzy implikace a neexistuje ustálená axiomatická definice, tedy fuzzy implikace je každá operace ve tvaru $\underset{\cdot}{\to}\colon \langle 0,1\rangle^2 \to \langle 0,1\rangle$, která se na $\{0,1\}^2$ shoduje s implikací klasické logiky. Níže jsou uvedeny její tři nejčastější typy:

$$\alpha \mathrel{\underset{\cdot}{\overset{R}{\to}}} \beta = \sup\left\{\gamma : \alpha \mathbin{\underset{\cdot}{\wedge}} \gamma \le \beta\right\} \tag{11}$$

$$\alpha \mathrel{\underset{\cdot}{\overset{S}{\to}}} \beta = \underset{S}{\neg}\alpha \mathbin{\dot{\vee}} \beta \tag{12}$$

$$\alpha \mathrel{\underset{\cdot}{\overset{Q}{\to}}} \beta = \underset{S}{\neg}\alpha \mathbin{\dot{\vee}} (\alpha \mathbin{\underset{\cdot}{\wedge}} \beta) \tag{13}$$

Lze si snadno vyzkoušet, že všechny tři typy fuzzy implikací vedou v případě $\{0,1\}^2$ ke klasické implikaci známé z Booleovy algebry. Různé implikace jsou odvozeny od různých konjunkcí. Podrobněji v textu [6].

2.5.6 Fuzzy biimplikace

Fuzzy *biimplikace*, neboli ekvivalence je odvozená od *fuzzy implikace* vztahem:

$$\alpha \dot{\leftrightarrow} \beta = (\alpha \dot{\rightarrow} \beta) \dot{\wedge} (\beta \dot{\rightarrow} \alpha).$$

Poznámka: Pojem *fuzzy ekvivalence* označuje další speciální relaci, proto preferujeme používání pojmu *biimplikace*.

2.5.7 Agregační operátory

Ve fuzzy logice je často třeba operace agregace, neboli sdružení názoru několika expertů. Tato operace nemá analogii v Booleovské algebře. Příklady agregačních operátorů jsou fuzzy průměry. Budeme uvažovat operace s aritou ≥ 2. Tento přístup popisuje Klir v knize [3].

Dá se dokázat, že zobrazení h, které každé n-tici hodnot z $\langle 0, 1\rangle$ $(n \geq 2)$ přiřadí číslo z $\langle 0, 1\rangle$ v souladu s následujícími podmínkami:

$$h(0, ..., 0) = 0, \quad h(1, ..., 1) = 1, \tag{14}$$

$$(\forall i = 1, ..., n : \alpha_i \leq \beta_i) \Rightarrow h(\alpha_1, ..., \alpha_n) \leq h(\beta_1, ..., \beta_n), \tag{15}$$

$$h \text{ je spojité.} \tag{16}$$

Pokud zobrazení splňuje také podmínky (17) a (18), nazývá se *fuzzy průměr* (anglicky averaging operator).

$$\text{Pro každou permutaci } p \text{ čísel } 1, ..., n \text{ je } h(\alpha_{p(1)}, ..., \alpha_{p(n)}) = h(\alpha_1, ..., \alpha_n), \tag{17}$$

$$\forall \alpha \in \langle 0, 1\rangle : h(\alpha, ..., \alpha) = \alpha. \tag{18}$$

Například všechny spojité fuzzy konjunkce a disjunkce splňují (14) – (17) a tak jsou agregačními operátory, navíc standardní konjunkce a disjunkce splňují i (18) a jsou tedy *fuzzy průměry* [6].

2.6 Logika

Fuzzy logika je v mnoha ohledech velmi podobná klasické logice. Logika je věda zkoumající lidské uvažování a způsob vyvozování závěrů. Klasická logika nám v mnoha případech stačí pro vyjadřování, ale fuzzy logika nám nabízí aparát, který je mnohem bližší lidskému uvažování. Největší význam přináší fuzzy logika v oblasti fuzzy řízení, kde napodobujeme jednoduchá pravidla, kterými se řídí člověk, odborník. Nyní je fuzzy řízení velmi úspěšné v mnoha odvětvích po celém světě, ale myšlenka vícehodnotové logiky pochází už od dob Aristotela. V průběhu 19. století prošla obrovským vývojem a dokonce někteří čeští autoři (např. Petr Hájek) [8] významně v této oblasti přispěli. Dále si definujme formuli fuzzy výrokové logiky:

- každá atomická formule je formule,

- nulární o je formule,
- jsou-li φ a ψ formule pak $(\varphi \wedge \psi)$ a $(\varphi \to \psi)$ jsou formule,

kde atomy značíme malými písmeny, logické spojky s aritou takto: nulární o jako nepravdivý výrok, binární konjunkce, binární implikace. Závorky někdy vynecháváme, když nehrozí nejednoznačnost. Další spojky lze odvodit od předchozích. Například negaci odvodíme jako $\neg\varphi = \varphi \to o$. Systém axiomů, které zavedl Hájek v textu [8]:

$$
\begin{array}{ll}
(B1) & (\varphi \to \psi) \to [(\psi \to \chi) \to (\varphi \to \chi)], \\
(B2) & (\varphi \bigwedge \psi) \to \varphi, \\
(B3) & (\varphi \bigwedge \psi) \to (\psi \bigwedge \varphi), \\
(B4) & [\varphi \bigwedge (\varphi \to \psi)] \to [\psi \bigwedge (\psi \to \varphi)], \\
(B5a) & [\varphi \to (\psi \to \chi)] \to [\varphi \bigwedge \psi) \to \chi], \\
(B5b) & [(\varphi \bigwedge \psi) \to \chi] \to [\varphi \to (\psi \to \chi)], \\
(B6) & [(\varphi \to \psi) \to \chi] \to [((\psi \to \varphi) \to \chi) \to \chi, \\
(B7) & o \to \varphi.
\end{array}
$$

odpovídá třídě fuzzy logik. Podobně jako v základní logice je součástí syntaxe fuzzy logiky jediné odvozovací pravidlo zvané *modus ponens*:

Z platnosti formulí $\varphi, \varphi \to \psi$ vyplývá platnost formule ψ.

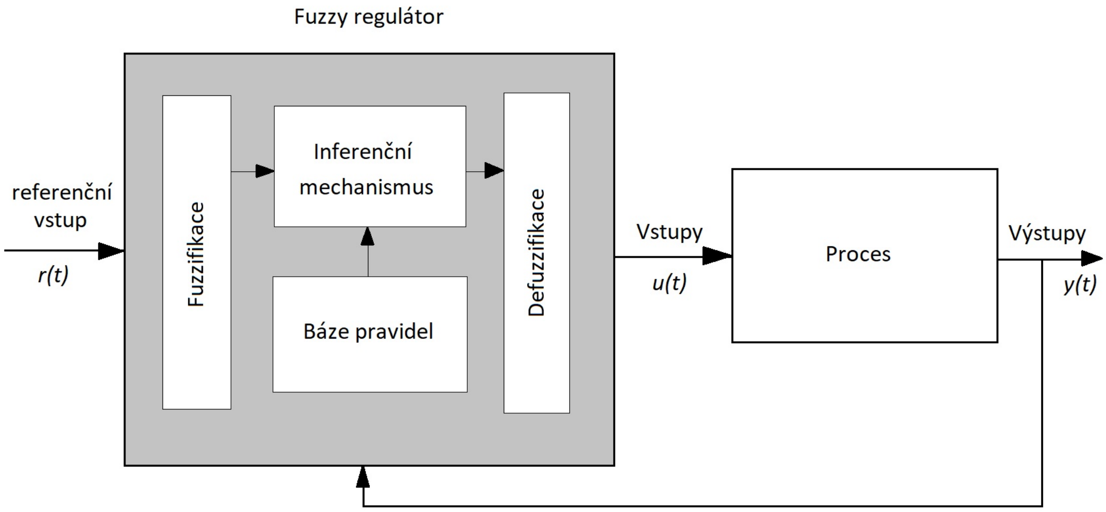
Za φ, ψ můžeme dosadit libovolné formule. Platnost jiných formulí vyvozujeme pomocí axiomů (nebo formulí, které vznikly z axiomů pomocí substituce) a odvozovacího pravidla *modus ponens*. Definujeme dokazatelnost formulí:

- formule, která vznikla z axiomu substitucí, je dokazatelná,
- jsou-li $\varphi, \varphi \to \psi$ dokazatelné formule, pak formule ψ je dokazatelná.

Posloupnost dokazatelných formulí, z nichž poslední je dokazované tvrzení se nazývá *logický důkaz*. Další typy fuzzy logik a podrobněji v knize [6].

3 Fuzzy regulátor

Hlavní cíl fuzzy řízení je řízení složitých technických procesů pomocí lidských zkušeností. Původně byly metody řízení pouhým překladem zkušeností do sady řídících pravidel. Dnes se používá inženýrský přístup, kdy se regulátor ladí, dokud není jeho chování dostatečně nezávislé na lidském chování. Klasické (ne fuzzy) řídící systémy jsou navrhované pomocí modelů. Inženýrské znalosti jsou potřebné k tomu, aby model odpovídal skutečnému procesu, jenž je velmi zjednodušován kvůli abstrakci. Existuje však jednak mnoho procesů, které lze řídit bez modelu, a dále mnoho procesů, jež mohou řídit jen lidé, nikoliv klasické systémy. Například lidé s řidičským průkazem dokáží řídit auto bez modelu činnosti [9].



Obrázek 7: Schéma fuzzy regulátoru [5]

Teoretický základ pro fuzzy řídící systémy položil článek „Outline of a New Approach to the Analysis of Complex Systems and Decision Processes“, napsaný v roce 1973 autorem Lotfi A. Zadehem. Za nejdůležitější součást všech fuzzy regulátorů jsou považována inferenční pravidla, která jsou v článku popsána [4]. Klíčovou roli ve vývoji regulátorů sehrál E. Mamdani, který vytvořil laboratorní model parního stroje řízeného fuzzy regulátorem [10]. Holmblad a Østergaard v roce 1982 navrhli první industriální aplikaci, a to řízení cementové pece v Dánsku [11]. V následujících letech se fuzzy řízení stalo populární v Japonsku, kde vytvořili řadu různých aplikací. Mnoho inženýrů stále nemá důvěru ve fuzzy řízení, ale situace se pomalu mění. V Japonsku měly koncem 80. let velký vliv na odbornou veřejnost populární články v médiích. Přispěly ke změně názoru na fuzzy řídící systémy jako na systémy levné, které lze jednoduše navrhnout a které mají lepší výkon než tradiční řídící systémy. Obecně se to ovšem tvrdit nedá, vždy záleží na konkrétním systému. Fuzzy řízení se tedy postupně stává uznávaným řídícím paradigmatem, a pravděpodobně bude hrát důležitou roli i v budoucnu [9].

Nejvýznamnější aplikací fuzzy logiky a také tím co ji proslavilo je fuzzy řízení. V mnoha případech je člověk nahrazen automatem, protože je to levnější a přesnější. V dnešní době již automat počítá úlohy pomocí diferenciálních či diferenčních rovnic, na které člověk nestačí. Fuzzy logika přináší výhodu tím, že umožňuje popis velmi podobný lidskému vyjadřování, a tak jsou pravidla lehce srozumitelná a dobře se s nimi pracuje [6].

Příklad úlohy, kterou lze obtížně popsat pomocí rovnic, ale poměrně intuitivně v termínech fuzzy logiky, je regulace otáček motoru auta. Řidič dobře zná základní pravidla, podle kterých ovlivňuje otáčky pro správnou funkci motoru.

Běžně při poklesu otáček řidič „přidá plyn“. Jiná situace žádá, aby řidič zařadil nižší stupeň rychlosti. U starších aut může být důvodem nízkých otáček studený motor a tak je třeba upravit režim karburátoru. Spojkou se udržují otáčky při rozjezdu vozidla. Všechna tato pravidla lze použít jen za příslušných podmínek (předpokladů), není například vhodné regulovat nízké otáčky spojkou, když je zařazený pátý rychlostní stupeň. Pomocí rovnic by se tato pravidla popisovala velmi složitě, ale ve větách typu „jestliže–pak“ jednoduše sestavíme bázi pravidel (anglicky rule base), na které bude fungovat fuzzy regulátor. Předpoklady a příslušné reakce budou popsány pomocí fuzzy množin. Vstupní hodnoty tvoří podmnožinu X aritmetického vektorového prostoru $\mathbb{R}^n$, $n \in \mathbb{N}$. Výstupní hodnoty tvoří podmnožinu $Y \subseteq \mathbb{R}^m$, $m \in \mathbb{N}$. Báze pravidel lze vyjádřit implikacemi typu

- jestliže $x \in A_1$, pak $y \in B_1$,
- jestliže $x \in A_2$, pak $y \in B_2$,
- ...

Množinu A_i nazýváme předpoklad neboli antecedent (anglicky antecedent, premise), množinu B_i nazýváme sukcedent (anglicky succedent, consequent) [6]. První množina vyjadřuje míru, jak dalece vyhovuje pravidlu, druhá vyjadřuje míru, jak moc je výstup žádoucí podle pravidla. Přirozeně se nabízí chápat bázi pravidel jako konjunkci implikací, neboť pravidla mají být splněna současně. Tedy relace $R \in F(X \times Y)$ lze definovat vzorcem

$$\mu_R(x,y) = \underset{\cdot\ i}{\bigwedge} \left(\mu_{A_i}(x) \overset{R}{\underset{\cdot}{\rightarrow}} \mu_{B_i}(y) \right).$$

Konjunkce odpovídá některé fuzzy konjunkci a implikace odpovídá reziduované implikaci. Takto definované zobrazení z X do Y reprezentuje bázi pravidel. Mnohem častěji se ale setkáváme s jiným přístupem a tím je tzv. Mamdaniho-Assilianův regulátor (také Mamdaniho regulátor), který zavádí relaci jako disjunkci konjunkcí, vzorcem:

$$\mu_R(x,y) = \dot{\bigvee}_i \mu_{A_i}(x) \underset{\cdot}{\wedge} \mu_{B_i}(y),$$

Tento přístup eliminuje problémy s časovou složitostí a nespojitými implikacemi, na které často narážel předchozí přístup. Mamdaniho-Assilianův regulátor uplatňuje spojité operace a je výpočetně méně náročný. Lze dokázat [6], že pokud předpoklady tvoří disjunktní pokrytí množiny vstupů, pak oba postupy vrací stejné výsledky; v praxi však obvykle netvoří disjunktní pokrytí a tak se výsledky většinou liší. Váhy jednotlivých pravidel se upravují podle různých vstupů a tak výstupy plynule přechází mezi hodnotami danými příslušnými pravidly. Výstupy, které jsou ve tvaru fuzzy množin se příliš nehodí v praxi pro řízení, a tak se dále defuzzifikují na konkrétní hodnotu. Existuje více možností, jak z fuzzy množiny získat přesnou hodnotu. Lze například vypočítat souřadnice těžiště pod grafem funkce příslušnosti, nebo zvolit bod s nejvyšším stupněm příslušnosti. Když je bodů více, můžeme vybrat jejich těžiště.

Takagiho-Sugenův regulátor je obecnější varianta Mamdaniho-Assilanova regulátoru [12]. Neomezuje sukcedent na fuzzy množinu, ale nabízí na jeho místo libovolnou funkci vstupu, kterou bývá většinou jiný regulátor. Lze takto zajistit, že první báze pravidel pouze specifikuje vhodný regulátor pro konkrétní situaci. Důležitý aspekt při návrhu fuzzy regulátoru je vhodně reprezentovat jednotlivé podmínky pomocí fuzzy množin. V další fázi se řešení optimalizuje například zkušebním provozem nebo sledováním experta. Zajímavou možnost poskytuje kombinace s neuronovými sítěmi či genetickými algoritmy [6].

3.1 Návrh

Návrh fuzzy regulátoru je klíčový v oblasti fuzzy řízení. Návrh spočívá v určení vstupních a výstupních fuzzy množin, a odvození pravidel řízení [12]. Obecně existují 4 hlavní zdroje pro získání báze pravidel [13].

1. Zkušenosti operátora – experta. Například příručka operátora cementové pece [11]. Často se pomocí vhodných dotazů na operátora, dle jeho zkušeností, sestaví báze pravidel.

2. Činnost operátora. Sledováním rozhodování a činností operátora je taktéž možné obdržet pravidla pro řízení.

3. Fuzzy model procesu. Pravidla mohou být chápána jako inverzní model procesu, který řídíme. Získání takových pravidel je tedy možné pomocí inverze fuzzy modelu. Tato metoda je omezena pouze na procesy, u kterých je návrh modelu možný.

4. Strojové učení. Neuronové sítě jsou jednou z dalších možností vytvoření báze pravidel.

Při návrhu řešení složitého řídícího problému pracuje inženýr automatizace podle obvyklého postupu. Příkladem takového problému je tempomat v automobilu, který umožňuje automobilu regulovat vlastní zrychlení, aby udržel konstantní rychlost zadanou řidičem vozu. Možné řešení konstrukce tempomatu je přídání ovladače (součástky), který zjišťuje rychlost vozidla z tachometru a umožňuje regulovat plyn tak, aby byla dosažena požadovaná rychlost. Tempomat musí přesně reagovat na změny sklonu vozovky, změny typu vozovky, případný čelní vítr či na různou celkovou hmotnost nákladu. Nejprve se inženýr podrobně seznámí s detaily problému a určí cíle. Poté typicky sestaví matematický model, který použije pro návrh regulátoru. Dále provádí simulace, pomocí kterých testuje výkon. Nakonec implementuje regulátor například pomocí mikroprocesoru. Během simulací nebo po implementaci je možné, že z výkonnostních důvodů je nutná malá úprava, nebo i výrazná změna návrhu a další přepracování. Fuzzy regulátory jsou jen jedna z možných metod používaných při návrhu řídících regulátorů. Klasické PID (proportional-integral-derivative) přístupy vytváří model popsaný diferenciálními rovnicemi. Hlavní rozdíl v návrhu fuzzy regulátorů je snaha o intuitivní porozumění procesu a následné předání znalostí přímo regulátoru bez použití složitých rovnic. V příkladu tempomatu je možné od běžného řidiče získat slovní popis pravidel, která používá pro udržování konstantní rychlosti. Jedno z možných pravidel je "Pokud je aktuální rychlost nižší než požadovaná, sešlápni víc pedál plynu". Při složitějších problémech je nutné bližší porozumění nebo znalost zkušeného operátora. Tato slovní pravidla jsou základem pro fuzzy řízení, obdobně jako jsou diferenciální rovnice základem pro klasické, konvenční řízení [5].

V mnohých případech je návrh modelu velmi složitého procesu či následná implementace příliš komplikovaná a fuzzy řízení je zde možnou alternativou.

3.2 Jednotlivé části regulátoru

Fuzzy regulátor se skládá ze čtyř hlavních částí:

1. Báze pravidel, která obsahuje všechny informace pro rozhodování.
2. Inferenční modul vyhodnocuje, která pravidla se vztahují k aktuálnímu stavu, a odvozuje fuzzy akční zásah.
3. Fuzzifikace převede vstupy regulátoru na fuzzy množiny, aby byly v příhodném tvaru ke zpracování pomocí pravidel.
4. Defuzzifikace vydá jedinou hodnotu jako výstup ze závěrů inferenčního modulu, často pomocí metody těžiště.

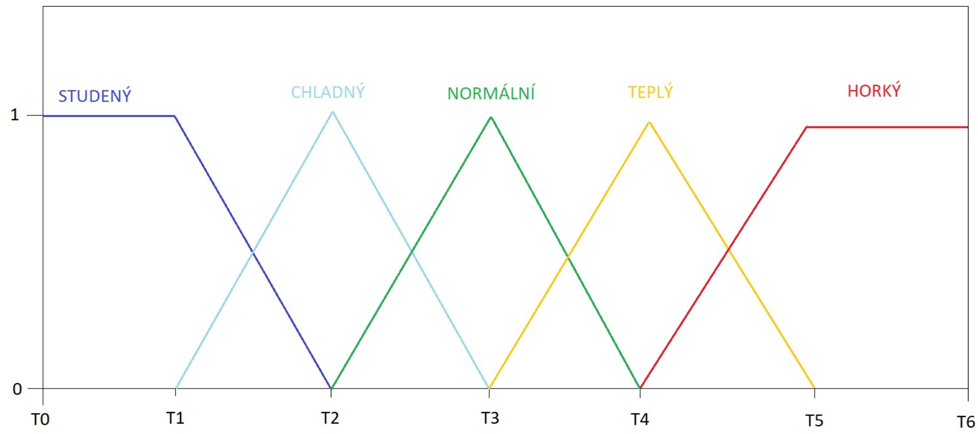
3.2.1 Fuzzifikace

Převedení číselných vstupů na fuzzy množiny pomocí jazykových proměnných se nazýva fuzzifikace.

Jazykové proměnné Pojem zavedl Lotfi A. Zadeh ve článku [14]. Každá proměnná je určena trojicí $\langle X, U, R(X)\rangle$, kde X je jméno proměnné, U je univerzum a $R(X)$ je obor hodnot, který je podmnožinou U. Například IQ je proměnná nabývající hodnot v univerzu

$$U = \mathbb{N} \text{ a } R(IQ) = \{1, 2, ..., 200\}.$$

Obor hodnot $R(X)$ může být také fuzzy množina v U. Pak nazýváme X fuzzy proměnnou. Každý prvek nenáleží pouze jedné množině, ale může být prvkem více množin a znázorňujeme, jakým stupněm jim přísluší [15]. Například z reálné proměnné *cena výrobku* vznikne po fuzzifikaci jazyková proměnná, která má pět atributů: *malá, spíše malá, střední, spíše vyšší, vysoká*. Další příklad je jazyková



Obrázek 8: Funkce příslušnosti pro *teplotu*

proměnná *výška*, která nabývá hodnot: *vysoký, velmi vysoký, velmi velmi vysoký, poměrně vysoký, ne příliš vysoký, celkem vysoký, víceméně vysoký.*

Nejprve určíme vstupní proměnné báze $x_1, ..., x_n$, výstupní proměnnou y a k nim příslušné jazykové proměnné $X_1, ..., X_n, Y$ tak, aby pokrývaly celé univerzum. Například proměnná *teplota* definována na univerzu [10,50]°C může nabývat nespočetně mnoho hodnot. Jazyková proměnná, která přísluší této proměnné má konečně mnoho hodnot, například: studený, chladný, normalní, teplý, horký, viz obrázek 8.

Univerzum Univerzum obsahuje všechny hodnoty, kterých může daná proměnná nabývat. Volba univerza je nutná před definicí funkcí příslušnosti. Například pravidlo „Jestliže je chyba negativní a změna chyby je pozitivní, pak výstup je 0“, má proměnné *chyba* a *změna chyby*, u kterých je třeba zvolit univerza.

Funkce příslušnosti Každý prvek v univerzu přísluší určitým stupněm, třeba i 0, fuzzy množině. Funkce příslušnosti mohou být různých tvarů a to nejčastěji trojúhelníkové, ale také tvaru trapezoidu nebo zvonu. Zvolený tvar funkce příslušnosti je obvykle méně důležitý, v praxi se ukazuje, že výsledky jsou srovnatelné. Důležitější při návrhu je počet křivek a jejich umístění pro optimální pokrytí univerza. Běžně se používá tři až sedm funkcí.

Existují jednoduchá pravidla [16], která dokáží zjednodušit návrh funkcí příslušnosti: například zvolit funkce ve tvaru trojúhelníků (viz sekce 2.2). Všechny funkce pro konkrétní vstup či výstup by měly být identické trojúhelníky. Krajní funkce budou pak pravoúhlé lichoběžníky. Dále, vzájemné překrytí sousedních funkcí by mělo být alespoň 50%. Tedy každá hodnota univerza přísluší alespoň dvoum funkcím.

3.2.2 Báze pravidel

Báze pravidel je množina pravidel daného tvaru:

$$\text{JESTLIŽE } x_1 \text{ je } A_{j1} \text{ a} \cdots \text{a } x_n \text{ je } A_{jn}, \text{ PAK } y \text{ je } B_j,$$

kde $x_1, ..., x_n, y$ jsou proměnné s možnými hodnotami v množinách $X_1, ..., X_n, Y$ a kde $A_{j1}, ..., A_{jn}, B$ jsou jazykové výrazy jako „vysoká", „dlouhá" apod [7].

Báze pravidel může používat jednu a více vstupních (výstupních) hodnot. Rozlišujeme SISO (single input single output) a MIMO (multi input multi output) problémy. Typicky SISO problém udržuje řídící signál podle chyby signálu, mnohdy regulátor potřebuje chybu i změnu chyby. I tak problém zařazujeme do kategorie SISO [17]. Pravidla jsou ve tvaru „jestliže – pak" [4]. Příklady pravidel: Jestliže je rychlost vysoká, pak dráha potřebná k zastavení je dlouhá. Proměnná rychlost může nabývat hodnot od 0 po 220 km/h, konkrétně u fuzzy množin *malá*, *střední*, *velká*. Proměnná dráha může nabývat hodnot od 0 po 300 m, pomocí fuzzy množin *krátká*, *střední*, *dlouhá*. Každé pravidlo ve fuzzy systému má stupeň pravdivosti, nakolik se hodí ke aktuální situaci. Každý závěr (druhá část pravidla) má stejný stupeň pravdivosti jako předpoklad (první část pravidla). Příklad báze pravidel pro model prodeje výrobku na základě výše spotřebitelských cen:

Jestliže je cena vysoká, pak bez prodeje.

Jestliže je cena spíše vyšší, pak malý prodej.

Jestliže je cena střední, pak střední prodej.

Jestliže je cena spíše malá, pak velký prodej.

Jestliže je cena malá, pak velký prodej.

Pravidla mohou obsahovat jeden i více předpokladů. Například: Obsluha je perfektní nebo jídlo je výborné, pak spropitné je vysoké. Obdobně mohou pravidla mít několik závěrů. Jestliže je teplota vysoká, pak teplé vody je ubráno, studené vody je přidáno.

3.2.3 Inferenční modul

Inferenční modul představuje algoritmus, který umožňuje na základě hodnot vstupních proměnných x_i odvodit hodnotu výstupní proměnné y. Hodnotou y může být i fuzzy množina v Y, ze které se metodou defuzzifikace vypočítá konkrétní hodnota v Y [7].

Inference se skládá ze dvou hlavních částí: hledání shody (anglicky matching) a inferenční krok (anglicky inference step) [5].

Hledání shody je první část, kdy určíme, na kolik jsou jednotlivá pravidla relevantní pro konkrétní situaci danou konkrétními vstupy. Nejprve přiřadíme jednotlivým vstupům předpoklady pravidel a poté určíme, která pravidla mají shodu. Dále můžeme uvažovat důležitost jednotlivých pravidel. Například když přiřadíme pravidlu stupeň důležitosti 0,1, pak není dané pravidlo příliš významné.

Odvozování závěrů z relevantních pravidel a vstupů nazýváme *inferenční krok*. Jsou dvě možnosti provedení kroku. První varianta spočívá v určení vyvozených fuzzy množin, neboli množin do kterých spadá výstup. Tento postup je použitý v sekci 5.2.1. Druhá možnost je určení jedné celkové vyvozené fuzzy množiny.

Fuzzy inferenční metody jsou rozdělené na přímé a nepřímé. Mamdani a Sugenův přístup jsou přímé metody, liší se v tom, jak získávají výstupní hodnoty.

Nejpoužívanější inferenční mechanizmus je Mamdaniho metoda. V roce 1975 profesor Ebrahim Mamdani na Londýnské univerzitě postavil jeden z prvních fuzzy regulátorů. Byl to systém pro řízení parního motoru (steam engine and boiler combination). Jako bázi pravidel použil zkušenosti operátorů. Mamdaniho fuzzy výpočet se provádí ve čtyřech částech: fuzzifikace, vyhodnocení pravidel, spojení výstupů pravidel a defuzzifikace.

Mamdaniho přístup lze ukázat na jednoduchém příkladu problému se dvěmi vstupy: financování projektu a personál projektu, a jedním výstupem: risk. Mějme bázi pravidel skládající se ze tří jednoduchých tvrzení:

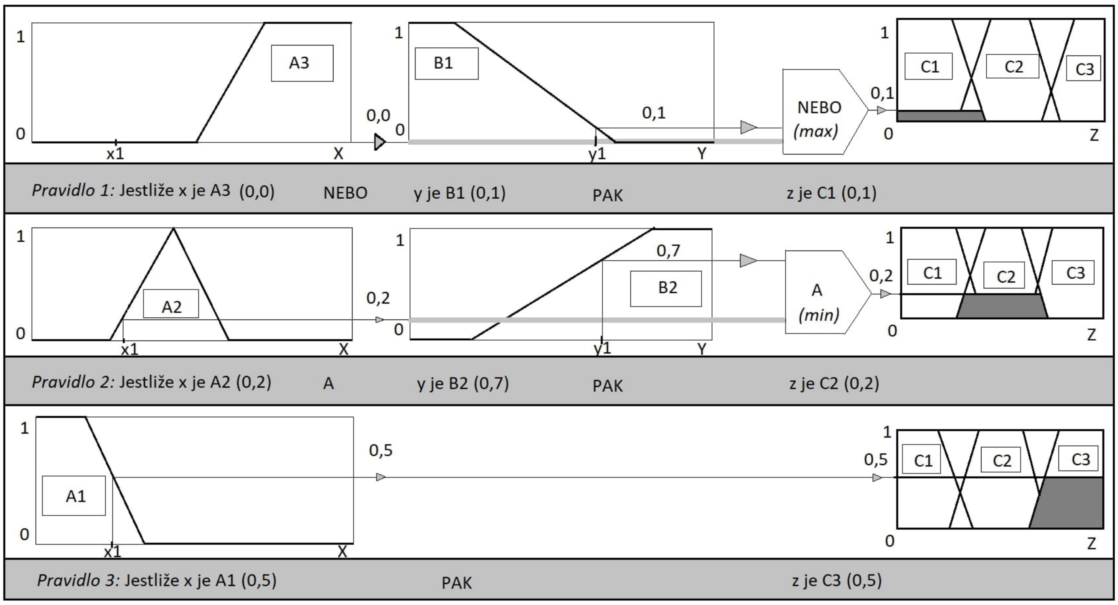
1. JESTLIŽE finance projektu jsou postačující (A3) NEBO personálu projektu je málo (B1), PAK risk je nízký (C1).
2. JESTLIŽE finance projektu jsou střední (A2) NEBO personálu projektu je mnoho (B2), PAK risk je normální (C2).
3. JESTLIŽE finance projektu jsou nedostačující (A1), PAK je risk vysoký (C3).

Prvním krokem je fuzzifikace. Vezmeme konkrétní vstupní hodnoty a určíme stupeň příslušnosti vstupů příslušným fuzzy možinám, zde například 0,5 (A1), 0,2 (A2), 0,1 (B1) a 0,7 (B2), kde množiny A znázorňují finance a množiny B personál. Dále vezmeme fuzzifikované vstupy a aplikujeme je na předpoklady (první části pravidel). Výsledné číslo, reprezentující stupeň, poté aplikujeme na závěr (druhou část pravidla).

Michio Sugeno navrhl použít *fuzzy singleton* jako funkci příslušnosti. Fuzzy singleton je fuzzy množina definovaná na jediném prvku univerza. Sugenův přístup je velmi podobný Mamdaniho inferenci, liší se v druhé části pravidel, kde používá funkci závislou na vstupech. Nejčasteji volí konstantu jako závěr (druhou část pravidla), tím zajišťuje, že výstup každého pravidla je právě konstanta.

3.2.4 Defuzzifikace

Fuzzy logika pro převod jazykových proměnných do reálných proměnných užívá různých transformačních postupů. Existuje mnoho metod defuzzifikace, neboli



Obrázek 9: Inference

získání jedné konkrétní hodnoty (anglicky *crisp*) z vyvozených fuzzy množin či jedné celkové vyvozené fuzzy množiny, podle zvoleného inferenčního postupu. Jedny z často používaných jsou metody maxima nebo těžiště.

Metoda těžiště Jedna z metod defuzzifikace, kdy fuzzy čísla nahrazujeme reálnými čísly. Anglicky *center of gravity* (COG).

$$y_q^{crisp} = \frac{\sum_{i=1}^{R} b_i^q \int y_q \mu_{B_q^i}(y_q) dy_q}{\sum_{i=1}^{R} \int y_q \mu_{B_q^i}(y_q) dy_q},$$

kde R je počet pravidel, b_i^q je střed plochy funkce příslušnosti B_q^p, která je spojená s vyvozenou množinou B_q^i pro i-té pravidlo a $\int_{y_q} \mu_{B_q^i}(y_q) dy_q$ označuje plochu pod $\mu_{B_q^i}(y_q)$ [5].

Ostatní metody *Center-average* je jiná možná metoda.

$$y^{crisp} = \frac{\sum_i b_i \mu_{p\check{r}edpoklad_{(i)}}}{\sum_i \mu_{p\check{r}edpoklad_{(i)}}},$$

kde b_i je střed plochy funkce příslušnosti a $\mu_{p\check{r}edpoklad_{(i)}}$ jsou hodnoty stupně příslušnosti předpokladů pro všechna i. Další možné metody deffuzifikace jsou například *středový bod středního intervalu* (MOM), *medián* a *očekávaná hodnota fuzzy čísla*. Další metody v knize [5].

3.3 Výhody fuzzy řízení

Pomocí fuzzy logiky jsme schopni převést sadu jazykových pravidel od zkušeného operátora na jejich matematické protějšky. Hlavní výhoda je usnadnění práce návrhu řídícího systému. Dále může fuzzy řízení přinést přesnější reprezentaci toho, jak systém reálně pracuje. V případě problémů, u kterých je návrh modelu příliš složitý, nabízí fuzzy řízení jednoduchou a flexibilní alternativu. Systém v *bázi* obsahuje *pravidla*, která jsou srozumitelná a pro zkušeného operátora jednoduše formulovatelná. *Báze pravidel* a *funkce příslušnosti* je možné definovat přímočaře, zjednodušují celkový návrh a také je snadné je po čase pozměnit v případě potřeby.

4 Aplikace fuzzy logiky v různých odvětvích

Fuzzy logika může být přínosná pro širokou škálu odvětví, například ve vědě, strojírenství, medicíně nebo i v podnikání a managementu. Od svých počátků v 70. letech minulého století zaujala mnohé odborníky, kteří zkoumali její užitek ve svém daném oboru. Příkladů a literatury popisující použití je mnoho a zde uvádím pouze úzký výběr napříč některými odvětvími.

Mezi prvními nápady na využití patří klasifikace (třídění) podle vzoru a také rozhodování. Na obou myšlenkách spolupracoval autor revolučního článku o fuzzy logice Lotfi A. Zadeh. Autoři článku [18] z roku 1970 popsali rozhodování, které používá dynamické programování, jehož cíl a omezení byla dána fuzzy množinami [19].

4.1 Fuzzy regulátor

Další velmi významnou aplikací je fuzzy regulátor. Myšlenku regulátorů založených na bázi pravidel ve tvaru „jestliže–pak“ popsal L. A. Zadeh ve článku roku 1972 [20]. První regulátor sestrojili Ebrahim H. Mamdani spolu s jeho žákem Sedrakem Assilianem v laboratoři na Londýnské univerzitě [10]. Jejich regulátor byl použit pro řízení jednoduchého parního stroje a v mnoha pokusech překonal rychlost klasického regulátoru při dosažení požadovaných řízených hodnot. V roce 1993 publikoval Ebrahim H. Mamdani článek, ve kterém líčí vlastní zkušenost s návrhem a důvody použití právě fuzzy přístupu.

4.2 Řízení cementové pece

První komerční využití přišlo v Dánsku roku 1980, kde byl vytvořen a použit fuzzy regulátor pro řízení cementové pece [11]. Do té doby řídili pec operátoři, kteří podstoupili osmitýdenní výcvik, aby získali potřebné znalosti a požadované pochopení postupu. Pro firmu to znamenalo veliké výdaje a uvažovala o nahrazení operátorů pomocí regulátorů. Návrhu strojového řízení se ujal Lauritz

P. Holmblad, inženýr z Dánské technické univerzity. Proces řízení pece se ukázal být příliš komplexní pro použití klasického regulátoru. Vhodnější se ukázalo fuzzy řízení, které vychází ze zkušeností operátora na rozdíl od modelování procesu. Pro návrh báze pravidel byla použita příručka pro operátory, obsahující 27 slovních pravidel, která byla jednoduše převedena do požadovaného tvaru. Lauritx P. Holmblad s dřívějším žákem Jens-Jorgen Østergaardem spolu navrhli fuzzy regulátor podle Mamdaniho vzoru, který ušetřil firmě čas potřebný pro školení operátorů a také snížil spotřebu paliva.

4.3 Řízení metra a fuzzy boom v Japonsku

Další senzační posun přineslo fuzzy řízení metra v Japonském městě Sendai. V roce 1979 začali japonští inženýři Seiji Yasunobu a Shoji Miyamoto pod záštitou firmy Hitachi a jejich výzkumného centra vyvíjet fuzzy regulátor, který řídil zrychlení, zpomalení a brždění vlaku. V roce 1987 došlo k nasazení fuzzy regulátoru a ukázalo se, že snižuje spotřebu energie o 10%, mnohem přesněji brzdí na požadované místo, a také nabízí mnohem plynulejší a pohodlnější jízdu pro cestující. Metro mělo dva režimy, řízení rychlosti a řízení zastavení, které měly na starost dva příslušné regulátory, které spolu komunikovaly. Po obrovském úspěchu fuzzy řízení ve městě Sendai byl do provozu nasazen obdobný regulátor i v Tokiu. V Japonsku se fuzzy řízení stalo velmi populární a v následujících letech bylo vyvinuto mnoho komerčně úspěšných výrobků založených na fuzzy logice. Dokonce byla Lotfi A. Zadehovi v roce 1989 udělena prestižní cena firmy Honda, která slouží k ocenění technologie, jež napomáhá k rozvoji lidské civilizace. Mezi aplikace fuzzy řízení vyvinuté v následujících letech v Japonsku patří řízení průmyslových továren, výtahů ve výškových budovách, dopravy v rozsáhlých městech a také různé funkce v automobilech, například převodovka, brzdy a tempomat. Největší komerční význam mělo nasazení fuzzy řízení do běžných spotřebičů, které jsou založené na znalostech zkušeného uživatele, jako jsou chytré pračky, videokamery se stabilizací obrazu, vařiče rýže, ledničky, klimatizace a mnohé další. Jeden z nejkomplexnějších fuzzy regulátorů navrhl Michio Sugeno pro řízení helikoptéry bez pilota pomocí slovních instrukcí, které byly posílány bezdrátově ze země do regulátoru umístěného v helikoptéře [21]. Bylo prokázáno, že helikoptéru je schopen řídit začátečník bez rozsáhlých znalostí dynamiky letu vrtulníku. Období od poloviny 80. do poloviny 90. let v Japonsku, které se označuje jako „fuzzy boom“ [22], vedlo k založení několika institucí pro výzkum aplikací fuzzy logiky a k celkové změně náhledu na fuzzy logiku dokonce i v Americe, kde na MIT zkoumali její výhody a přínos oproti klasickému řízení [23].

4.4 Soft computing

Později se začalo zkoumat spojení fuzzy logiky s jinými přístupy. V letech kolem roku 1990 začal vývoj spojení fuzzy logiky s *neuronovými sítěmi*, které by potenciálně zvětšilo možnosti aplikace fuzzy logiky. Neuronové sítě mohly zlepšit přizpůsobitelnost a umožnit učení se funkcí příslušnosti z příkladů, a fuzzy logika přinesla flexibilitu a robustnost. Další hybrid je spojení fuzzy logiky s *genetickými algoritmy*, který také rozšiřuje možnosti aplikací. Lotfi A. Zadeh souhrně pojmenoval tato spojení „soft computing“ což zastřešuje spojení výpočetních metod, zejména *fuzzy logiky*, *neuronových sítí*, a *genetických algoritmů* [24]. Tyto metody nepracují odděleně, ale doplňují se a využívají nepřesnost k dosažení robustnosti, nižší ceny řešení a lepšího souladu s realitou. Lotfi A. Zadeh ve článku popisuje, jak lidé dokáží zaparkovat automobil ze znalosti aktulání a konečné polohy, přičemž jsou tyto polohy určeny nepřesně a se stoupající přesností se geometricky zvětšuje náročnost úkolu. Od té doby se *soft computing* stala samostatnou oblastí výzkumu a staví na ní mimo jiné *inteligentní systémy*, což jsou umělé systémy schopné provádět vysoce komplexní úkoly podle vzoru lidského přístupu.

4.5 Fuzzy řízení a rozhodování

Fuzzy řízení byla jedna z prvních oblastí, kde se objevily výhody tohoto nového principu. Oproti klasickému řízení, které je založené na matematickém modelu procesu, je fuzzy řízení založené na matematické reprezentaci znalostí zkušeného operátora, jak bylo popsáno dříve v práci v kapitole 3. Dále je v části 5.2.1 popsán jednoduchý regulátor Mamdaniho typu, který slouží ke stabilizaci inverzního kyvadla.

Fuzzy rozhodování popsané ve článcích [18] a [25] využívá právě fuzzy logiku k umožnění slovního popisu cílů, omezení rozhodovacího problému a jejich zpracování pomocí fuzzy množin. Oblast rozhodování se obvykle dělí na dvě hlavní části: rozhodování jednotlivce a rozhodování skupiny.

4.6 Biologie

Pro analýzu biologických dat, která jsou zpravidla daleko rozsáhlejší než běžná data, je třeba naprosto odlišného matematického přístupu [26]. Při vzniku bioinformatiky, která sbírá, ukládá a třídí biochemická a biologická data pomocí počítačových programů, ve spolupráci s Projektem lidského genomu, vznikla obrovská databáze obsahující struktury všech lidských genů. Pro lepší využití tak objemných dat se ukázala fuzzy logika jako výhodná, protože dokáže analyzovat stupně podobnosti a také charakterizovat a porovnávat vhodně struktury.

5 Programátorská dokumentace

Aplikace vizualizuje průběh výpočtu fuzzy regulátoru, který řídí inverzní kyvadlo. Uživatel je seznámen s problémem inverzního kyvadla a dále proveden jednotlivými fázemi získání výstupu z daných vstupů. Aplikace nabízí jednoduché grafické rozhraní s jediným oknem a intuitivním ovládání tlačítky dále a zpět. Výpočet je rozdělen do základních fází, které jsou vysvětleny a dále zobrazeny za pomocí grafických transformací. Uživatel si může vyzkoušet výpočet pro různé vstupy a v každé fázi se vrátit a změnit je. Při pohybu tlačítky vpřed se spouští grafické transformace znovu, s případně změněnými vstupy, a naopak při pohybu zpět se po předchozím dokončení pouze zobrazí finální stav.

5.1 Použité technologie a knihovny

Aplikace je psaná v jazyce Java na platformě Java SE Development Kit 8, pro uživatelské prostředí je použita platforma JavaFX.

Jazyk Java je objektově orientovaný jazyk, jehož syntaxe je založená na programovacích jazycích typu C. Je velmi rozšířený a oblíbený pro svou přenositelnost mezi platformami. Přenositelnost je zaručená tím, že místo strojového kódu vytváří kompilátor bajtový kód, který je nezávislý na systému zařízení. Virtuální stroj Javy, Java Virtual Machine (JVM), je běhové prostředí, které se postará o spuštění bajtového kódu na libovolném zařízení, které disponuje virtuálním strojem. Před spuštěním libovolného programu v Javě tedy stačí implementovat modul JVM na daném počítači.

V dřívějších verzích Javy byla rychlost programu omezená interpretací, dnes je možné provádět kompilaci za běhu – Just in time compilation (JIT), kdy není kód přímo interpretován, ale před prvním provedením je dynamicky zkompilován do strojového kódu daného počítače. Pro potřeby této aplikace je rychlost plně dostačující, neboť aplikace je zaměřená hlavně na uživatelské rozhraní s menším množstvím výpočtů.

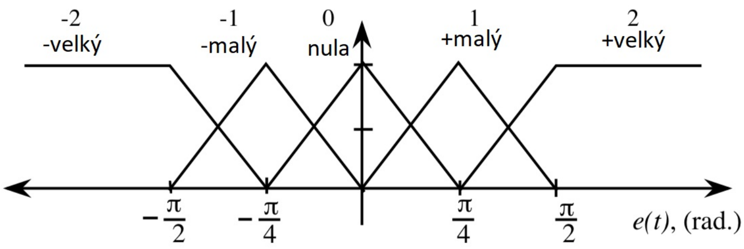
JavaFX je moderní framework pro tvorbu vizuálně bohatých okenních aplikací, slouží primárně pro vývoj tzv. RIA aplikací (Rich Internet applications). JavaFX přináší podporu obrázků, videí, hudby, grafů, CSS stylů a dalších technologií, které umožňují vytvářet desktopové a mobilní aplikace, nebo také webové aplikace. JavaFX byla součástí Javy ve verzi 8, dnes od verze 11 tomu tak není a je třeba ji doinstalovat zvlášť, nebo ji použít jako knihovnu.

Pro vytváření vzhledu byl použit jazyk FXML, což je jazyk založený na bázi XML, kterým je možné vytvořit strukturu vzhledu a tak ji oddělit od aplikační logiky. FXML soubor popisuje pouze vzhled aplikace a přísluší mu controller, který je psán v jazyce Java a stará se o funkčnost. Pro vytvoření základního vzhledu byl využit grafický designer JavaFX Scene Builder, který usnadňuje tvorbu FXML souboru a tedy celkového vzhledu.

Pro stylování komponent byly použity kaskádové styly (CSS), což je jazyk pro popis způsobu zobrazení elementů napsaných ve značkovacích jazycích jako jsou HTML nebo XML.

5.2 Struktura projektu

Aplikace se skládá ze tříd, souborů definujících vzhled a také jednoho CSS souboru upravujícího vzhled komponent. Ve složce images jsou obsažené potřebné grafické prvky.



Obrázek 10: Funkce příslušnosti pro první vstup (*chyba*) inverzního kyvadla [5]

5.2.1 Implementace fuzzy regulátoru řídící inverzní kyvadlo

Calculation je třída představující jádro výpočtu fuzzy regulátoru, který řídí inverzní kyvadlo. Obsahuje vstupy a výstup regulátoru, bázi pravidel, metody pro fuzzifikaci a defuzzifikaci a také metody pro získávání informací, které jsou použity v grafické prezentaci.

- Třída calculation obsahuje atributy `x1` a `x2`, které představují dva vstupy regulátoru: *chyba* a *změna chyby*. Jsou přijímány v radiánech a radiánech za sekundu. Vstupy mohou nabývat po řadě hodnot z intervalů $[-\pi, \pi]$ a $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$. *Chyba* označuje velikost úhlu mezi aktuální polohou kyvadla a požadovanou vertikální. *Změna chyby* představuje směr a rychlost pohybu kyvadla.

- Funkce příslušnosti jsou trojuhelníkového tvaru a pro jednotlivé vstupy i výstup je jich definovaných 5. Rozdělují tedy interval možných hodnot na množiny které mají názvy *-velký, -malý, nula, +malý, +velký*, viz obrázek 10. Proměnná `mf1[i] (mf2[j])` značí hodnotu funkce příslušnosti spojené s prvním (druhým) vstupem a jazykové proměnné `i (j)`. Poznámka: při implementaci nahradíme značení $-2, -1, 0, 1, 2$ používané pro znázornění jazykových proměnných *-velký, -malý, nula, +malý, +velký* za obvyklejší indexování: $0, 1, 2, 3, 4$. Tedy `i` $= 0, 1, 2, 3, 4$ a `j` $= 0, 1, 2, 3, 4$. Pro výpočet stupně příslušnosti `i`-té funkce příslušnosti použijeme pomocnou funkci, která používá jednoduché rovnice pro reprezentaci trojúhleníkových funkcí příslušnosti. Například zavoláním pomocné funkce `mf1(i, x1)`, kde `i` $= 2$ a `x2` $= \frac{\pi}{16}$ dostaneme výsledek 0,75, což znamená, že hodnota $\frac{\pi}{16}$ patří fuzzy množině *nula* na 75%.

- Báze obsahuje 25 pravidel, pro všechny kombinace dvou vstupů, které jsou rozdělené na 5 funkcí ($5^2 = 25$.). Tabulkou lze jednoduše zaznačit všechna možná pravidla viz obrázek 11. Proměnná `rule[i,j]` představuje index jazykové proměnné výstupu pravidla: Jestliže `x1` je `i` a `x2` je `j`, pak `y` je

síla u		změna chyby $\dot{e}$				
		-2	-1	0	1	2
chyba e	-2	2	2	2	1	0
	-1	2	2	1	0	-1
	0	2	1	0	-1	-2
	1	1	0	-1	-2	-2
	2	0	-1	-2	-2	-2

Obrázek 11: Báze pravidel pro inverzní kyvadlo [5]

`rule[i,j]`. `Rule[][]` obsahuje matici, která uchovává tabulku pravidel

$$\texttt{rule[][]} = \begin{bmatrix} 4 & 4 & 4 & 3 & 2 \\ 4 & 4 & 3 & 2 & 1 \\ 4 & 3 & 2 & 1 & 0 \\ 3 & 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

- `prem[i,j]` označuje, jakým stupněm z intervalu $[0,1]$ je pravidlo `rule[i,j]` aktivní. Pro pravidla která začínají „Jestliže `x1` je `i` a `x2` je `j`“, používáme k zjištění celkového stupně minimum ze stupňů obou částí spojenými spojkou „a“, viz třetí řádek ve zdrojovém kódu 1.
- `center[k]` označuje hodnotu, kterou má `k`-tá funkce příslušnosti uprostřed. Pro inverzní kyvadlo hodnota $k = 0, 1, 2, 3, 4$, jsou středy v místě, kde trojúhelníkové funkce nabývají maxima.
- `areaimp[k,h]` označuje plochu pod `k`-tou trojúhelníkovou funkcí příslušnosti výstupu, která byla useknutá ve výšce `h`. Plocha je počítána pomocnou funkcí.
- `imps[i,j]` označuje plochy pod funkcemi příslušnosti určenými výpočtem pomocí `areaimp[k,h]` z jednotlivých pravidel `rule[i,j]` a jejich stupňů `prem[i,j]`, viz čtvrtý řádek ve zdrojovém kódu 1.

Při implementaci jsem čerpala z knihy [5], která obsahuje rady a příklady pro implementaci fuzzy regulátorů. V případě, že by byla dána omezení na operační paměť, nebo při implementaci regulátoru pro reálné kyvadlo by bylo vhodné algoritmus optimalizovat. Zde je pro lepší pochopení a jednoduchost uveden v původní neoptimalizované verzi.

5.2.2 Ostatní třídy

RegulatorFX je třída, která reprezentuje celou aplikaci. Třída *dědí* z *javafx.application.Application* a obsahuje spustitelnou funkci *main*. Její účel je vytvoření *okna* aplikace, vzhledu první *scény* a jejich provázání.

```
Získání vstupních hodnot x1 a x2
Výpočet mf1[i] a mf2[j] pro všechna i, j
prem[i,j] = min(mf1[i],mf2[j]) pro všechna i, j
imps[i,j] = areaimp(rule[i,j],prem[i,j]) pro všechna i, j
num = 0, den = 0
for i = 0 to 4, j = 0 to 4
  num = num + imps[i,j] * center [rule[i,j]]
  den = den + imps[i,j]
end
return ucrisp = num / den
```

Zdrojový kód 1: Fuzzy regulátor pro inverzní kyvadlo, používající metodu těžiště pro defuzzifikaci [5]

Rule je pravidlo, které slouží pro vytvoření grafické reprezentace báze pravidel.

Controllers jsou ostatní třídy pro příslušné FXML dokumenty a řídí funkčnost jednotlivých scén. Vytváří animace a starají se o funkce tlačítek pro pohyb dále a zpět aplikací.

5.2.3 FXML soubory

Pro každou *scénu* je vytvořen FXML soubor, který definuje její vzhled. Každá scéna obsahuje název příslušné části výpočtu regulátoru a tlačítka dále a zpět. Všechny části výpočtu jsou nejprve vysvětleny a dále ukázány jednoduchou animací nebo tabulkou.

6 Uživatelská dokumentace

Aplikace je navržena s důrazem na jednoduché a intuitivní ovládání, z toho důvodu obsahuje malé množství ovládacích prvků. Tato kapitola obsahuje návod, jak aplikaci spustit a jak s ní pracovat.

6.1 Technické požadavky

Aplikace je určena pro operační systémy macOS, Windows i Linux, které mají nainstalovaný JVM verze 8. Aplikace byla testována na macOS Mojave 10.14 a Windows 10.

6.2 Spuštění

Aplikaci není třeba instalovat. Pro spuštění stačí mít nainstalovaný virtuální stroj Javy (JVM verze 8), který je dostupný na adrese: https://java.com/en/download/. Na uvedené webové stránce je třeba vybrat a nainstalovat verzi pro konkrétní operační systém. Soubor aplikace *Vizualizace.jar* následně stačí spustit.

6.3 Hlavní navigace

Každá stránka aplikace obsahuje tlačítka pro pohyb dále a zpět v aplikaci. Při kliknutí na tlačítko dále se spouštějí dostupné animace následující stránky a při kliknutí na tlačítko zpět se pouze zobrazí předchozí stránka.

6.4 Volba jazyka

Úvodní obrazovka obsahuje volbu jazyka, výchozí nastavení je čeština. Pomocí tlačítka *EN* je možné přepnout na anglickou verzi a pomocí tlačítka *CZ* zpět na českou.

6.5 Určení vstupních hodnot

V sekci *určení vstupních hodnot* je třeba zadat vstupy fuzzy regulátoru, vysvětlení jednotlivých vstupů je v předchozí sekci. Oba vstupy jsou přijímány v radiánech a lze je zadat do textových polí nebo vybrat pomocí posuvníku. Při zadání neplatné hodnoty je doplněna 0, při zadání hodnoty mimo možný interval je doplněna nejbližší platná hodnota.

6.6 Aktivní pravidla

V této sekci jsou uvedena aktivní pravidla, která jsou dána vstupními hodnotami. Do tabulky je možné klikat a měnit volbu řádků. Tím se mění aktivní pravidla a je možné si vyzkoušet jak se mění i slovní vyjádření pravidel.

6.7 Reference

V poslední sekci je navíc uvedené tlačítko *Zdroje*, které slouží pro zobrazení použitých zdrojů pro vytvoření aplikace.

Závěr

Práce představuje čtenáři fuzzy regulátor a objasňuje jeho vnitřní fungování.

V první kapitole byl na základě studia literatury zpracován úvod do fuzzy logiky, definovány pojmy a operace. Následující kapitoly byly věnovány fuzzy regulátoru a aplikacím fuzzy logiky. V další části této bakalářské práce se nachází programátorská dokumentace, kde jsou popsané použité technologie a celková struktura aplikace. Na závěr je uvedena uživatelská dokumentace, která obsahuje návod jak aplikaci spustit a jak s ní pracovat.

V rámci mé práce a jako její nedílná součást byla vyvinuta aplikace, která vizualizuje činnost fuzzy regulátoru inverzního kyvadla. V aplikaci je možné volit různé vstupy a sledovat, jak se mění jednotlivé fáze a výstup regulátoru. Aplikace by mohla být v budoucnu rozšířena o různé tvary a počet funkcí příslušnosti, o možnost měnit bázi pravidel a jiné metody defuzzifikace a jejich srovnání.

Conclusions

The thesis introduces a fuzzy controller and explains each of its key components.

The first chapter consists of an introduction to fuzzy logic with many important definitions. The next chapters were devoted to fuzzy control and various successful applications of fuzzy control and fuzzy logic. The following chapter contains a software documentation, describing the technology used to make the program and an overview of the programs structure. In the last chapter there is the user documentation with instructions how to run the program and how to work with it.

The program developed, which visualizes the way a fuzzy controller for an inverted pendulum works, was an important part of my thesis. In the program, it is possible to choose the input values and observe how it effects all the stages and output of the controller. In the future it would be possible to improve the program by adding different shapes and numbers of membership functions, by adding the option to change the rule base and adding more defuzzification methods to compare.

A Obsah přiloženého CD

Tato část stručně popisuje obsah přiloženého CD, tj. jeho závazné adresářové struktury, důležité soubory apod.

`bin/`

Aplikace Vizualizace spustitelná přímo z CD. Adresář obsahuje všechny JAR soubory, knihovny a další soubory potřebné pro bezproblémové spuštění.

`doc/`

Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky, apod.

`src/`

Kompletní zdrojové texty aplikace Vizualizace se všemi potřebnými zdrojovými texty, knihovnami a dalšími soubory potřebnými pro bezproblémové vytvoření spustitelných verzí programu.

`readme.txt`

Instrukce pro instalaci a spuštění programu Vizualizace, včetně všech požadavků pro jeho bezproblémový provoz.

Literatura

[1] WEAVER, Warren. *Science and complexity. In: Facets of systems science.* Springer, Boston, MA, 1991. p. 449–456.

[2] BREMERMANN, Hans J., et al. *Optimization through evolution and recombination.* Self-organizing systems, 1962, 93: 106.

[3] KLIR, George J.; YUAN, Baozung. *Fuzzy sets and fuzzy logic: theory and applications.* New Jersey: Prentice Hall PTR, 1995.

[4] ZADEH, Lotfi A. *Outline of a new approach to the analysis of complex systems and decision processes.* IEEE Transactions on systems, Man, and Cybernetics, 1973, 1: 28–44.

[5] PASSINO, Kevin M.; YURKOVICH, Stephen; REINFRANK, Michael. *Fuzzy control.* Menlo Park, CA: Addison-wesley, 1998.

[6] NAVARA, Mirko; OLŠÁK, Petr. *Základy fuzzy množin.* České vysoké učení technické, 2002.

[7] BĚLOHLÁVEK, Radim. *Olomoucký informatický korespondenční seminář OLINX, Matematika a strojové myšlení, 4. díl.* 2017. Dostupné z: https://soubor.inf.upol.cz/shares/5fbd2c.

[8] HÁJEK, Petr. *Metamathematics of fuzzy logic.* Springer Science & Business Media, 2013.

[9] ZIMMERMANN, Hans-Jürgen. *Fuzzy set theory—and its applications.* Springer Science & Business Media, 2011.

[10] MAMDANI, Ebrahim H.; ASSILIAN, Sedrak. *An experiment in linguistic synthesis with a fuzzy logic controller.* International journal of man-machine studies, 1975, 7.1: 1–13.

[11] HOLMBLAD, Lauritz P.; ØSTERGAARD, Jens-Jørgen. *Control of a cement kiln by fuzzy logic.* In: Readings in Fuzzy Sets for Intelligent Systems. Morgan Kaufmann, 1993. p. 337–347.

[12] SUGENO, Michio. *An introductory survey of fuzzy control.* Information sciences, 1985, 36.1–2: 59–83.

[13] LEE, Chuen-Chien. *Fuzzy logic in control systems: fuzzy logic controller. II.* IEEE Transactions on systems, man, and cybernetics, 1990, 20.2: 419–435.

[14] ZADEH, Lotfi Asker. *The concept of a linguistic variable and its application to approximate reasoning—I.* Information sciences, 1975, 8.3: 199–249.

[15] NOVÁK, Vilém. *Fuzzy množiny a jejich aplikace.* Státní nakladatelství technické literatury, 1990.

[16] HILL, G.; HORSTKOTTE, E.; TEICHROW, J. *TIL Shell User's Manual*. Togai InfraLogic Inc, 1990.

[17] JANTZEN, Jan. *Foundations of fuzzy control: a practical approach*. John Wiley & Sons, 2013.

[18] BELLMAN, Richard E.; ZADEH, Lotfi Asker. *Decision-making in a fuzzy environment*. Management science, 1970, 17.4: 141–164.

[19] BĚLOHLÁVEK, Radim; DAUBEN, Joseph Warren; KLIR, George J. *Fuzzy logic and mathematics: a historical perspective*. Oxford University Press, 2017.

[20] ZADEH, Lofti A. *A rationale for fuzzy control*. Journal of Dynamic Systems, Measurement, and Control, 1972, 94.1: 3–4.

[21] SUGENO, Michino, et al. *Development of an intelligent unmanned helicopter*. In: Proceedings of 1995 IEEE International Conference on Fuzzy Systems. IEEE, 1995. p. 33–34.

[22] FREIBERGER, M. Paul; MCNEILL, Daniel. *Fuzzy Logic: The Discovery of a Revolutionary Computer Technology—and How It Is Changing Our World*. New York: Touchstone, 1993.

[23] BERNARD, John A. *Use of a rule-based system for process control*. IEEE Control Systems Magazine, 1988, 8.5: 3–13.

[24] ZADEH, Lofti A. *Fuzzy logic, neural networks, and soft computing*. Communications of the ACM, 1994, 37.3: 77–85.

[25] KICKERT, Walter JM. *Fuzzy theories on decision making: A critical review*. Springer Science & Business Media, 1979.

[26] ZADEH, Lotfi A. *From circuit theory to system theory*. Proceedings of the IRE, 1962, 50.5: 856–865.

[27] ZADEH, Lotfi A. *A fuzzy-set-theoretic interpretation of linguistic hedges*. 1972.

[28] YASUNOBU, Seiji; MIYAMOTO, Shoji; IHARA, Hirokazu. *Method and device for stopping vehicle at predetermined position*. U.S. Patent No 5,018,689, 1991.

[29] NGUYEN, Hung T.; WALKER, Carol L.; WALKER, Elbert A. A *first course in fuzzy logic*. CRC press, 2018.

[30] ZADEH, Lotfi A. *Fuzzy sets*. Information and control, 1965, 8.3: 338–353.

[31] MAMDANI, Ebrahim H. T*wenty years of fuzzy control: experiences gained and lessons learnt*. In: [Proceedings 1993] Second IEEE International Conference on Fuzzy Systems. IEEE, 1993. p. 339–344.

[32] TAKAGI, Tomohiro; SUGENO, Michio. F*uzzy identification of systems and its applications to modeling and control*. In: Readings in Fuzzy Sets for Intelligent Systems. Morgan Kaufmann, 1993. p. 387–403.