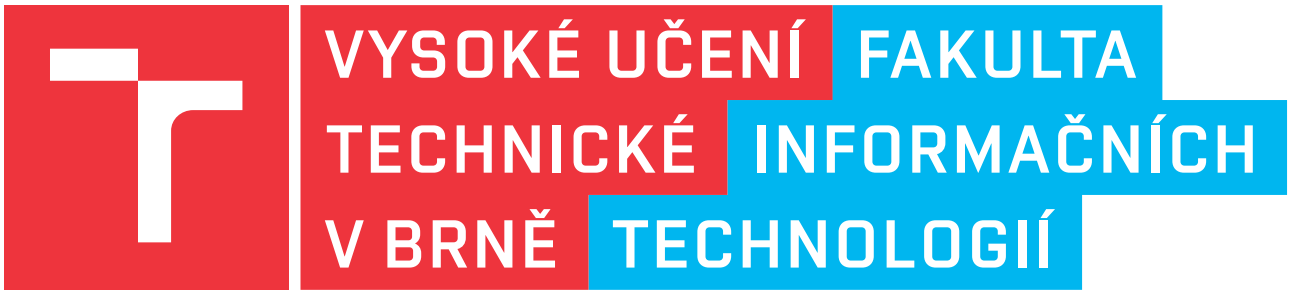


Demonstrace základních operací s fuzzy množinami (C/C++)

27. října 2017

Nikola Valešová (xvales02)

1 Úvod

Fuzzy logika je zobecněním klasické logiky, kterou rozšiřuje o možnost vyjádřit nebinární příslušnosti prvků do množin. Díky tomu je lépe schopna reflektovat skutečnost, jevy reálného světa, u kterých nejsme schopni přesně určit hranici oddělující prvky, které do dané množiny náleží, a prvky, jenž této množině nenáleží.

Tato práce má za cíl graficky demonstrovat význam základních operací nad fuzzy množinami. K tomuto účelu bylo vybráno pět nejznámějších typů funkce příslušnosti. Uživatel se tedy seznámí jak s demonstrovanými operacemi na fuzzy množinách, tak i s implementovanými typy funkce příslušnosti, přičemž aplikace umožňuje libovolně měnit typ obou vstupních fuzzy množin (nezávisle na sobě) a také číselné parametry odpovídajících funkcí příslušnosti.

2 Návrh aplikace

Demonstrátor fuzzy operací bude implementován jako aplikace s grafickým uživatelským rozhraním. Veškeré zadávání parametrů i získávání výstupů musí být tedy řešeno s využitím dostupných grafických prvků. Z pohledu uživatele bude důležité, aby se změny vstupních hodnot ihned zobrazily do výstupů a změnily se odpovídající grafy v okamžiku dokončení změn parametrů. Pro jednoduchost a přehlednost budou také všechny grafy (tedy se vstupními množinami i s odpovídajícími výsledky fuzzy operací) zobrazeny současně.

3 Popis implementovaných principů/metod

3.1 Typické funkce příslušnosti

Funkce příslušnosti fuzzy množiny A na množině X je definována jako zobrazení $m_A : X \to \langle 0, 1\rangle$, které všem prvkům $x \in X$ přiřazuje hodnotu z intervalu $\langle 0, 1\rangle$ udávající, s jakým stupněm příslušnosti $m_A(x)$ do dané množiny A náleží. V aplikaci jsou prezentovány nejčastěji se vyskytující typy těchto funkcí, jejichž popis implementace je obsahem této sekce. Pro všechny zmiňované funkce platí, že jejich vstupem jsou hodnoty parametrů dané funkce příslušnosti a jejich výstupem je vektor bodů požadované fuzzy množiny. Vztahy použité v této sekci byly převzaty ze zdroje [1].

3.1.1 Trojúhelníková

Trojúhelníková funkce příslušnosti je určena třemi parametry, a, b a c, přičemž musí platit nerovnost $a < b < c$. Jejich hodnoty určují x-ové souřadnice tří vrcholů pomyslného trojúhelníku, víme tedy, že fuzzy množině s trojúhelníkovou funkcí příslušnosti budou vždy náležet body $[a, 0]$, $[b, 1]$ a $[c, 0]$. Výpočet všech bodů dané množiny je určen vztahem: $triangle(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right)$ a v aplikaci se jeho implementace nachází v metodě `ComputeTrianglePoints`.

3.1.2 Lichoběžníková

Funkce příslušnosti lichoběžníkového tvaru je určena čtyřmi parametry, a, b, c a d, kde platí nerovnost $a < b < c < d$. Hodnoty těchto parametrů určují x-ové souřadnice čtyř vrcholů lichoběžníku. V tomto případě budou tedy fuzzy množině vždy náležet body $[a, 0]$, $[b, 1]$, $[c, 1]$ a $[d, 0]$. Obecný výpočet bodů

dané fuzzy množiny je potom určen vztahem: $trapezoid(x; a, b, c, d) = \max\left(\min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right)$, v aplikaci je realizován v metodě `ComputeTrapezoidalPoints`.

3.1.3 Gaussova

Gaussova funkce příslušnosti je dána dvěma parametry, c a σ. Parametr c udává x-ovou souřadnici středu dané množiny a σ má vliv na šířku, ovlivňuje tedy strmost bočních hran. Pro Gaussovu funkci přílušnosti platí vztah: $gaussian(x; c, \sigma) = e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2}$, pro jehož výpočet je v aplikaci možno zavolat metodu `ComputeGaussianPoints`.

3.1.4 Zvonková

Tvar zvonkové funkce příslušnosti je určen parametry a, b a c. Předpis pro výpočet souřadnic bodů této funkce příslušnosti je následující: $bell(x; a, b, c) = \frac{1}{1+|\frac{x-c}{a}|^{2b}}$, přičemž musí platit $a \neq 0$. V aplikaci jeho výpočet probíhá v metodě s názvem `ComputeBellPoints`.

3.1.5 Sigmoidální

Poslední implementovaná funkce příslušnosti je sigmoidální, jenž je definována dvěma parametry, a a c. První z parametrů má vliv na strmost funkce příslušnosti a jeho znaménko určuje, zda funkce příslušnosti na celém intervalu roste, nebo klesá. Druhý parametr udává x-ovou souřadnici inflexního bodu grafu. Pro získání souřadnic bodů množiny se sigmoidální funkcí příslušnosti lze využít vztah: $sigmoidal(x; a, c) = \frac{1}{1+e^{-a(x-c)}}$. V implementované aplikaci k tomuto účelu slouží metoda s názvem `ComputeSigmoidalPoints`.

3.2 Základní fuzzy operace

Implementované operace lze rozdělit na binární a unární, přičemž některé z nich mají obdobný význam jako tytéž operace v klasickém množinovém počtu. Oproti klasickým množinám se zde navíc vyskytují tři unární operace – koncentrace, dilatace a normalizace. Vztahy použité v této sekci byly přejaty ze zdroje [2].

3.2.1 Binární operace

Binární operace mají jako vstupní parametry dva vektory bodů, které reprezentují dvě fuzzy množiny, A a B. Jejich výstupem je potom vektor bodů, který je výsledkem aplikace požadované operace na vstupní vektory.

Průnik Operace průniku je počítána v metodě `ComputeIntersect`. Využívá vztahu pro výpočet průniku dvou fuzzy množin: $A \cap B = \{x | \min(m_A(x), m_B(x)), x \in U\}$, tedy přiřazuje každému prvku, který náleží oběma množinám, menší ze dvou stupňů příslušnosti. Prvky náležící pouze jedné z množin budou mít stupeň příslušnosti roven 0.

Sjednocení Sjednocení dvou množin je realizováno metodou `ComputeUnion`, kde je získáváno pomocí vztahu: $A \cup B = \{x | \max(m_A(x), m_B(x)), x \in U\}$. Prvky, které náleží oběma sjednocovaným množinám, získají vyšší z těchto dvou stupňů příslušnosti. Prvky náležící pouze jedné z množin budou mít přiřazeny stupeň příslušnosti rovný stupni příslušnosti v původní množině.

3.2.2 Unární operace

Unární operace mají na vstupu jediný vektor bodů, který představuje fuzzy množinu A či B. Výstupem unárních operací je i zde vektor bodů, který byl vypočten aplikací dané funkce na body vstupní fuzzy množiny.

Doplněk Výpočet unární operace doplněk množiny A, $\neg A$, vychází ze vztahu: $\neg A = \{x|(1 - m_A(x)),\ x \in U\}$ a je implementován v metodě `ComputeComplement`. Každému prvku dané množiny je tedy přiřazena „opačná“ hodnota stupně příslušnosti.

Koncentrace Koncentrace, $con(A)$, je implementována v metodě `ComputeConcentration` a je definována vztahem: $con(A) = \{x|m_A^2(x), x \in U\}$ – všem prvkům náležícím dané vstupní množině je přiřazena druhá mocnina původního stupně příslušnosti. Jelikož se jedná o hodnoty z intervalu $\langle 0, 1\rangle$, kromě krajních hodnot tohoto intervalu dochází vždy ke zmenšení dané hodnoty a pro fuzzy množinu jako celek se stávají hrany strmější.

Dilatace Operace dilatace, $dil(A)$, je definována vztahem: $dil(A) = \{x|\sqrt{m_A(x)}, x \in U\}$ a její implementace se nachází v metodě `ComputeDilation`. Dilatace je inverzní operací k operaci koncentrace.

Normalizace Pro výpočet normalizace, $norm(A)$, se užívá vztahu: $norm(A) = \left\{x|\frac{m_A(x)}{\max(m_A(y)))}, x, y \in U\right\}$. Její implementaci lze nalézt v metodě `ComputeNormalization`. Význam této operace je především pro fuzzy množiny, jejichž jádrem je prázdná množina. Aplikací této operace na danou množinu získáme plný rozsah stupňů příslušnosti a množina má tedy neprázdné jádro.

Support Aplikace byla dále doplněna o výpočet supportu množiny, $support(A)$, jehož výsledkem je podmnožina původní množiny definována jako množina všech prvků splňujících následující vlastnost: $support(A) = \{x|m_A(x) > 0, x \in U\}$. Její implementace je uvedena v metodě, která nese název `ComputeSupport`.

4 Implementační detaily

Pro realizaci demonstrátoru byl zvolen programovací jazyk C/C++ doplněný o grafické prvky poskytované knihovnou *Qt*[1] verze 5.5.1, která je dostupná také na referenčním serveru `merlin`.

Z její podmnožiny *QtWidgets*[2] byl využit například prvek *QLabel*, který slouží k zobrazení textu, element *QLineEdit*, což je editovatelné textové pole, jímž je zajištěno zadávání vstupních parametrů, a prvek *QComboBox* umožňující výběr jedné z několika nabízených možností, který se zdál být ideální volbou pro řešení výběru typu funkce příslušnosti pro každou ze vstupních množin.

K vizualizaci výsledků byly využity prostředky dostupné z knihovny *QCustomPlot*[3], konkrétně grafický prvek *QCustomPlot*, který umožňuje vytvářet a upravovat grafy, jejich osy i popisky. Soubory této knihovny jsou (po osobní domluvě) přibaleny jako součást projektu pro zajištění bezproblémového překladu. Jedná se o soubory `qcustomplot.cpp` a `qcustomplot.h`, které se

[1] https://www.qt.io

[2] http://doc.qt.io/qt-5/qtwidgets-index.html

[3] http://qcustomplot.com/

nachází ve složce *libs* a jejichž nejsem autorkou. Slouží ovšem pouze k zobrazování grafů z dat, které jsou objektům této třídy předávány mou aplikací.

Aby bylo možné realizovat složitější numerické operace, jako například $\sqrt{x}$, e^x a x^2, byly využity funkce poskytované knihovnou *cmath*[4].

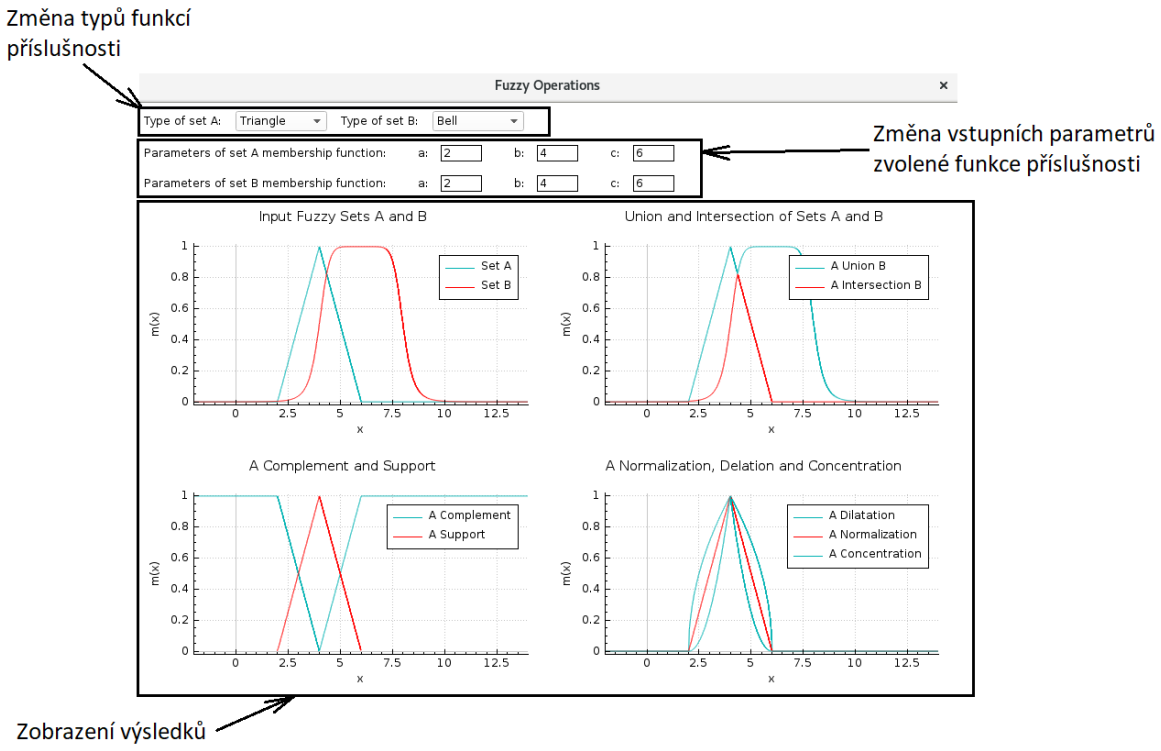
5 Překlad a spuštění aplikace

Aplikace je doplněna o soubor `Makefile`, který obsahuje veškeré příkazy potřebné pro překlad zdrojového kódu a vytvoření spustitelných souborů. Aplikaci lze tedy přeložit zadáním příkazu: `make`. Následně je možné aplikaci spustit pomocí příkazu: `make run`.

Pro správné zobrazení grafického uživatelského rozhraní na fakultním serveru `merlin` pro připojení z prostředí unixového typu je třeba se na fakultní server připojit s povolením grafických prvků přepínačem `-X` pomocí příkazu: `ssh -X [login]@merlin.fit.vutbr.cz`. Pro připojení na zmíněný server z OS řady Windows je třeba nejprve nainstalovat lokální X11 server a poté provést konfiguraci nástroje `Putty`.

6 Ovládání aplikace

Aplikace je ovládána skrze grafické uživatelské rozhraní, které je možno logicky rozdělit na dvě části – vstupní a výstupní. Jak je možno vidět na obrázku 1, grafické prvky pro zadávání vstupních hodnot se nachází v levé horní části okna aplikace, zatímco výstupní část pokrývá celý zbytek okna.



Obrázek 1: Popis ovládání aplikace

[4] http://www.cplusplus.com/reference/cmath

Nejvýše jsou zobrazeny dva rozbalovací seznamy pro výběr typů funkcí příslušnosti, které budou použity pro výpočet bodů vstupních fuzzy množin A a B. Pod těmito seznamy je možno nalézt editovatelná pole s příslušnými popisky. V nich jsou nastavovány číselné parametry pro zvolené typy funkcí příslušnosti. Po ukončení editace každého z polí jsou změny ihned propsány na výstup, není zde tedy třeba tlačítek. Druhá, výstupní část okna zabírá největší část aplikace a sestává ze čtyř grafů.

Graf vlevo nahoře zobrazuje vstupní fuzzy množiny vytvořené na základě výše nastavených parametrů. Graf vpravo nahoře obsahuje výsledky binárních fuzzy operací, sjednocení a průniku. Dolní dva grafy vizualizují výsledky unárních operací nad vstupní fuzzy množinou A. Vlevo dole jsou zobrazeny operace komplement a support, vpravo operace dilatace, normalizace a koncentrace. Všechny grafy jsou opatřeny nadpisem a legendou pro snadnou čitelnost zobrazovaných výsledků.

7 Závěr

Výsledkem tohoto projektu je aplikace schopná na základě požadavků získaných od uživatele graficky zobrazovat následující operace na fuzzy množinách: průnik, sjednocení, doplněk, dilataci, normalizaci, kontrakci a support. Aplikace je navíc schopna demonstrovat pět základních typů funkce příslušnosti, kterými jsou trojúhelníková, lichoběžníková, Gaussova, zvonková a sigmoidální.

Reference

[1] Fuzzy Logic Membership Function. *Welcome to research hubs* [online], [cit. 2017-10-14]. Dostupné z: http://researchhubs.com/post/engineering/fuzzy-system/fuzzy-membership-function.html.

[2] ZBOŘIL, František. *Fuzzy množiny, fuzzy logika, fuzzy inference* [online], Brno, 2016 [cit. 2017-10-14]. Dostupné z: https://www.fit.vutbr.cz/study/courses/SFC/private/16sfc_9.pdf.