KÖZÖS FEJLESZTŐI KERETRENDSZER FUZZY SZABÁLY INTERPOLÁCIÓS MÓDSZEREKHEZ

COMMON FUZZY RULE INTERPOLATION FRAMEWORK LIBRARY FOR DEVELOPERS

Krizsán Zoltán[*]

ABSTRACT

In case of applications running in real environments one often applies a sparse rule base. although there is not any developers' library supporting such applications. Behind the scenes, the models of such real applications use Fuzzy Rule Interpolation methods to determine the consequents taking into consideration some known rules. Johanyák implemented a new common framework, which includes several FRI methods. However, the disadvantage of his solution is the cumbersome usage owing its demand on the availability of a Matlab system. This article introduces an easy-to–extend and quick framework that support the development of fuzzy systems and which can be used from any of the popular programming languages (e.g. C, C++, Java, C#).

1. BEVEZETÉS

A fuzzy rendszerek a következtetéseket a szabály bázis alapján állítják elő. A klasszikus fuzzy rendszerek megkövetelik a bemenetek teljes mértékű lefedését szabályokkal. Ez azt jelenti, hogy minden megfigyelésre biztosan létezik legalább egy szabály antecedens, így egy érvényes következtetés számolható. A teljes lefedettség nagy számú szabályt igényel. A gyakorlatban ezért gyakori a ritka szabálybázisú rendszer, mert vagy nem áll még rendelkezésre elegendő információ, vagy nem lenne hatékony a teljesen fedő rendszer.

Jelen pillanatban még nincs olyan programozói keretrendszer, melynek segítségével a ritka szabálybázisú rendszerekben a fuzzy interpolációs módszerek használhatóak. Ennek a hiánynak köszönhető, hogy a ritka szabálybázisú interpolációs módszereket nem használják a gyakorlatban.

Ha a fejlesztendő rendszer ritka szabálybázist igényel, akkor a programozónak implementálnia kell a kiválasztott módszert a használni kívánt programozói nyelven. Ha egy másik alkalommal ismét ugyanazt a módszert szeretné használni, de egy másik programozó nyelvben, akkor újfent implementálnia kell azt azon a másik nyelven is. Minden egyes módszert minden kívánt nyelven implementálni és tesztelni kell. Ezek ráfordítást, erőforrásokat igényelnek.

Ha a fejlesztő nem rendelkezik a fuzzy rendszerek területén ismerettel, tapasztalattal, akkor egy MATLAB szoftver és az FRI MATLAB Toolbox szükséges, hogy kipróbálja és összevesse a rendelkezésre álló módszereket. Ha a projekt nem követeli meg a MATLAB rendszer meglétét, akkor ez felesleges költséget, és a MATLAB szoftver rendszer felhasználói szintű ismeretét igényli.

Azon rendszerek esetén, amelyek gyors beavatkozást, reakciót igényelnek alacsony szintű implementációk szükségesek. Ez azt jelenti, hogy az algoritmusokat abban a rendszerben kell implementálni (lehetőleg c++, vagy C nyelven), ahol használják. Minden közbenső réteg, beágyazott környezet drasztikus teljesítmény veszteséggel járhat. A már létező FRI MATLAB Toolbox a MATLAB speciális környezetére írt, annak speciális program nyelvén készült. Használata vagy MATLAB alatt, vagy beágyazva egy C++ alkalmazásba lehetséges. A beágyazás esetén az alkalmazásunkhoz kell csatolni a MATLAB natív könyvtárait, és az FRI MATLAB Toolbox instrukciói egy interpretált környezetben futnak. Az eredmények kiértékelése nem elég gyors a beágyazott futtatás esetén.

Elsődleges célunk egy olyan modul vagy komponens kifejlesztése, amely ritka szabálybázisú alkalmazások esetén is ad eredményt, könnyen használható, több nyelvet is támogat (lehetőleg azonos felülettel), MATLAB és ingyenes matematikai szoftverekben is használható, elegendően gyors a valós idejű szabályozásokhoz, vezérlésekhez.

Első lépésként Windows és Linux operációs rendszereket támogatjuk, törekedünk a manapság népszerű programozó nyelvek egységes támogatására (C, C++, Java, C#, Python).

A cikk első részében megvizsgáljuk milyen módszerek állnak rendelkezésre, melyeket kellene támogatnia a rendszerünknek, majd megvizsgáljuk, hogy milyen feltételeket kell figyelembe vennie. Ezután a rendszerünk felépítését, kiterjesztésének lehetőségét ismertetjük, végül a keretrendszer használatának módját.

[*] *egyetemi tanársegéd, Miskolci Egyetem, ALT*

2. FRI MÓDSZEREK, KÖZÖS JELLEMZŐI

A ritka szabálybázisú fuzzy rendszerekben történő következtetésre kifejlesztett módszereket két csoportba sorolhatjuk.

Az egy lépéses vagy direkt módszerek a következményt egy lépésben állítják elő. Ezen csoport meghatározó képviselői: a KH [1] melyet Kóczy és Hirota vezetett be, a MACI [2] (Tikk and Baranyi), a FIVE [3] (Kovács és Kóczy), az IMUL [4] (Wong, Gedeon, and Tikk), és VKK [5] (Vass, Kalmár and Kóczy) módszerek.

Találhatóak olyan módszerek is melyek nem egy, hanem két lépésben határozzák meg a következményt. Első lépésben egy ideiglenes szabályt hoz létre a megfigyelés helyén, majd második lépésben ezen ideiglenes szabály alapján meghatározza a következményt a megfelelő helyen megfelelő alakkal. Ezen eljárások lépéseit Baranyi és társai definiálták [6]. Ennek a családnak a módszerei az ST [7] (Yan, Mizumoto, and Qiao), az IGRV [8] Huang and Shen, , és a Jenei által javasolt [9] módszerek.

Ezen módszereket egyesítette Johanyák kifejlesztve egy MATLAB Fuzzy Toolbox-ot (FRI TB) [10]. Ez a MATLAB keretrendszer Matlab metódusok gyűjteménye. A keretrendszer letölthető a [11] helyről, és használható a „GNU General Public License" licensz szerint. MATLAB R14 alatt készült, így használható Windows és Linux rendszerek alatt is. Az FRI TB fis fájlt használ a fuzzy rendszer definiálására ugyanúgy, mint a Fuzzy Toolbox is, azonban a magában foglalt módszereknek köszönhetően tudja kezelni a hiányos szabálybázisú rendszereket is. Ezen keretrendszer segítségével olyan felhasználók is tudják használni az FRI módszereket, akik nem jártasak ezen a területen. A keretrendszer grafikus felületén megadhatóak a paraméterek, majd az eredményeket a MATLAB képes megjeleníteni. Ha a felhasználói rendszer futhat a MATLAB alól, akkor ez a keretrendszer ideális az eredmények előállítására.

Saját keretrendszerük kifejlesztésének alapjait ezen rendszer adja, azonban a mi implementációnkat mások kódjából fogják használni (nem grafikusan), így meg kell terveznünk egy egységes szerkezetet. A módszerek egységes kezeléséhez, helyes működésé érdekében meg kell vizsgálnunk azokat a közös vonásokat, feltételeket, melyeket minden FRI módszernek ki kell elégítenie.

Tikk és szerzőtársai [12] általános feltételeket definiáltak a fuzzy interpolációs módszerekkel szemben. Ezek azonban csak kívánalmak az egységesítés érdekében, nem kötelező jellegűek.

3. FRI PROGRAMOZÓI KERETRENDSZER

A keretrendszerrel szemben támasztott főbb követelmények a következők:

- *gyorsaság*, azaz a valós időben működő szabályzó, vezérlő alkalmazások válaszideje kellően kicsi lehet csak,
- *széleskörű támogatottság*: minél több projektben lehessen alkalmazni. Ez azt követeli, hogy több programozói nyelvből is lehessen használni. Célul tűztük ki a matematikai szoftverek támogatását is: MATLAB, Freemat.
- *egységes megközelítés:* A rendszerünk több módszert is támogat. Cél olyan felépítés kidolgozása, ami a módszerek között nem tesz különbséget, azonos felületen keresztül lehet azokat használni.

Az előbb említett követelmények miatt több formában is szállítjuk a keretrendszert: egy programozó könyvtár (C++ library), dinamikusan betölthető könyvtár (*.dll Windows esetén és *.lo Linux esetén) és beágyazó osztályokat a népszerűbb programozó nyelvekhez (C#, Java, Python), amelyek a dll-t használják.

Az egységes használat érdekében úgy terveztük meg az osztály hierarchiát, hogy bármely azonos típusú FRI módszert azonos felületen keresztül lehessen használni.

Két fajta használatot is támogatunk: *egyszerű* és *asszisztált*.

Az egyszerű használati mód azt jelenti, hogy az osztályokat közvetlenül lehet használni, azaz példányosítani, majd ezután metódusait hívni. Ebben az esetben a fejlesztő feladata és felelőssége létrehozni, tárolni és kellő időben törölni a fuzzy rendszert és az azokhoz kapcsolódó segéd objektumokat. A rendszerek tárolására a dinamikus vector struktúra javasolt, mert ekkor az új elemek hozzáadása, létezők törlése, megfelelő elemek kiolvasása megoldott a már létező operátorok segítségével.

Az asszisztált használati mód pedig azt jelenti, hogy a fejlesztő csak kéréseket küld a rendszerünknek egy új rendszer létrehozását, vagy létező rendszer törlését kérve.

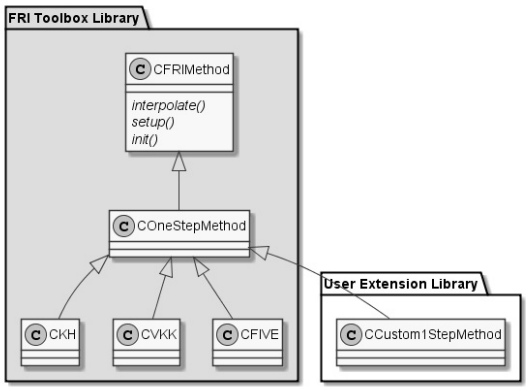
Tikk és szerzőtársai által meghatározott tulajdonságokból csak az első támogatható, hiszen a megvizsgált módszerek több tagja sem elégíti ki a többi feltételt. Az FRI módszerek implementációi kivételeket idéznek elő közvetlen használat esetén. Asszisztált használat esetén a metódusok visszatérési értéke képviseli a műveletek sikeres voltát. A sikeresen futott függvények nulla értékkel térnek vissza, a sikertelenek pedig egy hibakóddal, ami egy negatív érték. Minden hibának saját hibakódja van az egyértelmű hibakezelés érdekében.

3.1. A keretrendszer szerkezete

A keretrendszer magja C++ nyelvben íródott annak gyorsasága, és az objektum orientált volta miatt. Az objektum orientáltság lehetőséget nyújt az újrahasználható és az átdefiniálható program elemek

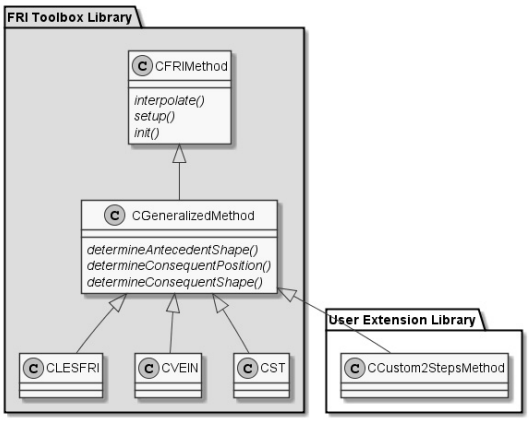
használatára, így az előre megírt kódot a végfelhasználó testre szabhatja, újra értelmezheti bizonyos elemeit.
Mivel a rendszer osztályok halmaza, így egyes részeit is használhatja saját logikájának megvalósítására. A mi könyvtárunk támogatja mind az egy lépéses, mind a két lépéses módszereket. A rendszer definiálására ugyanúgy a FIS file-t használja, mint az FRI Matlab Toolbox, a kompatibilitás érdekében. Ha a végfelhasználó már használta a Matlab fuzzy toolbox-át, vagy a Johanyák által kifejlesztett FRI Matlab Toolbox- ot, akkor ugyanazokat a rendszert leíró fájlokat itt is használhatja. Miután a rendszerünk létrehozta megfelelő fuzzy rendszert, eltárolja azt. A fejlesztő nem férhet hozzá közvetlenül a belső objektumokhoz, csak hivatkozhat azokra minden használat alkalmával. A fuzzy rendszer létrehozása és törlése egyszerű metódus hívás, és nem memóriát kezelő operátor használat. Keretrendszerünk használatára ez utóbbi lehetőség javasolt.
Ha a keretrendszerünket egy C++ alkalmazásba kívánják használni mindkét módot választhatják, mert a könyvtár tartalmazza mind az osztályok hierarchiáját, mind a menedzselő kód is. A dll használata esetén azonban csak az asszisztált mód lehetséges, kompatibilitási és biztonsági okokból. Ha a burkoló osztályokat (wrapper) használják (C#, Java, vagy Python) akkor is csak asszisztált használat lehetséges, mert a különböző programozói nyelvek csak a C-ben megírt dll-eket támogatják.
Az egységes használat érdekében minden módszer osztálya közvetve vagy közvetlenül a `CFRIMethod` osztályból származik. Ez a közös ősosztály biztosítja az a különböző módszerek egységes interfészét. A felhasználó négy lépésben tudja használni az adott FRI módszer:

1. Rendszer létrehozás: Első lépésként a Fuzzy rendszert kell létrehoznia (Ennek szabálybázisa lehet ritka is.)
2. FRI módszer kiválasztása: Itt adja meg a felhasználó, hogy most éppen melyik módszert fogja a későbbiekben használni. A módszer neve alapján létrejön a megfelelő objektum a háttérben. Ez a lépés később többször is elvégezhető a `setup` metódus hívásával. Így lehetőség nyílik a program futása során módszert váltani.
3. FRI módszer inicializálása: A módszerek különbözősége miatt lehetőséget biztosítunk egyéb módszer függő paraméterek használatára, melyet az `init` metódussal lehet megadni.
4. Következmény lekérdezése: A használat fő lépése, mely során az adott megfigyeléshez tartozó konzekvens lekérdezhető. Az `interpolate` metódus hívásán keresztül az éppen kiválasztott FRI módszer meghatározza a következményt.



1. ábra. Egy lépéses FRI módszerek struktúrája

Az egylépéses FRI módszerek mindegyike a `COneStepMethod` osztályból származik és felül kell definiálni az `interpolate` metódust. Ha új egylépéses módszert szeretnénk létrehozni, akkor ezen szabályok szerint kell létrehozni, mint `CCustom1StepMethod` osztály az 1. ábrán.
A keretrendszer már tartalmaz néhány ilyen módszert, melyek KH (`CKK` osztály), VKK (`CVKK` osztály) és FIVE (`CFIVE` osztály) módszerek.
A kétlépéses FRI módszerek pedig `CGeneralizedMethod` osztályból származnak és felül kell definiálnia a `determineAntecedentShapes`, `determineConsequentPositions` és `determineConsequentShapes` metódusokat. Ha egy két egylépéses módszert szeretnénk bevezetni, akkor ezen szabályok szerint kell létrehozni, mint `CCustom2StepMethod` osztály a [2] ábrán.
A keretrendszerben már található néhány ilyen módszer, melyek LESFRI (CLESFRI osztály), VEIN (CVEIN osztály) és ST (CST osztály) módszerek.



2. ábra. Két lépéses FRI módszerek struktúrája

1.1. A keretrendszer használata

Az új keretrendszert mind programkódból, mind szoftverek (Matlab, FreeMat) alól lehet használni. C++ projektek esetén a meglévő osztályokat közvetlenül használhatjuk, valamint a legjobb teljesítmény is így érhető el, azonban ilyenkor több a fejlesztő feladata és felelőssége is. Minden más esetben a dll-en keresztül használhatjuk a keretrendszert.

A dll használata szinte minden mai programozói kódból megoldott csupán az adott környezet leírását kell tanulmányoznunk. Az egyszerű használat érdekében a legnépszerűbb nyelvekhez készítettünk beágyazó osztályokat (C#, Java, Python, VB.NET), amelyeket csak példányosítani kell, majd hívni a megfelelő metódusokat.

Elsődleges célunk között szerepelt a MATLAB támogatása, annak népszerűsége, és gazdag matematikai metódusgyűjteménye miatt. A MATLAB lehetőséget biztosít dll betöltésére, és a benne levő globális függvények használatára. Ehhez a header és a dll fájl, valamint a loadlibrary és calllib függvények használata szükséges.

A MATLAB mellett az ingyenes matematikai rendszerek támogatása is fontos számunkra, ezért a FreeMat ingyenesen használható, kis méretű szoftvert támogatjuk elsőként, amely letölthető a [13] helyről. A MATLAB-hoz hasonló módon itt is lehetőség van a dll-ben levő függvények használatára. Itt az import függvényt kell használnunk, hogy a külső függvényeket belsőként tudjuk elérni.

2. KÖVETKEZTETÉS

A ritka szabálybázisú rendszerek esetében is használható, módszereket implementálta Johanyák MATLAB rendszerben, azonban a gyakorlati alkalmazása valós környezetben nehézkes.

Ezen cikk által bevezetett Fuzzy szabály interpolációs programozói keretrendszert (FRI Toolbox library), amely szabadon elérhető programozói könyvtár és dll. Ezen új megoldás használatával valós szabályzó és vezérlő alkalmazások fejleszthetőek, köszönhetően a gyors válaszidőnek és a könnyű használatnak. Ezt a több módszert is implementáló keretrendszert könnyen használhatjuk bármely manapság népszerű nyelv esetén, és nincs szükség a MATLAB komplex matematikai rendszerre.

Azoknak a fejlesztőknek javasoljuk, akik valós környezetben futó rendszereket fejlesztenek.

A rendszerünk továbbfejlesztéseként több FRI metódust szeretnénk integrálni valamint egy benchmark rendszert kifejleszteni, amelyben az implementált módszereket össze lehet vetni minta adatok alapján.

5. KÖSZÖNETNYILVÁNÍTÁS

A bemutatott kutató munka a TÁMOP-4.2.1.B-10/2/KONV-2010-0001 jelű projekt részeként az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

6. IRODALOM

[1.] Kóczy L. T. and Hirota K., Rule interpolation by α-level sets in fuzzy approximate reasoning, BUSEFAL, vol. 46, no. Automne, pp. 115–123, 1991.

[2.] D. Tikk and P. Baranyi, "Comprehensive analysis of a new fuzzy rule interpolation method," IEEE Trans. On Fuzzy Systems, vol. 8, no. 3, pp. 281–296, 2000.

[3.] S. Kovács and L. T. Kóczy, "Application of an approximate fuzzy logic controller in an agv steering system, path tracking and collision avoidance strategy," Tatra Mountains Math. Publ., vol. 16, pp. 456–467, 1999.

[4.] K. W. Wong, T. D. Gedeon, and D. Tikk, "An improved multidimensional α-cut based fuzzy interpolation technique," in Proc. of the Int. Conf. on Artificial Intelligence in Science and Technology (AISAT'00), V. Karri and M. Negnevitsky, Eds., Hobart, Tasmania, Australia, December, 2000, pp. 33–38.

[5.] G. Vass, L. Kalmár, and L. T. Kóczy, "Extension of the fuzzy rule interpolation method," in Proc. of the Int. Conf. on Fuzzy Sets Theory and its Applications (FSTA'92), Liptovsk´ y Jan, Slovakia, 1992, pp. 1–6.

[6.] P. Baranyi, L. T. Kóczy, and T. D. Gedeon, "A generalized concept for fuzzy rule interpolation," IEEE Trans. on Fuzzy Systems, vol. 12, no. 6, pp. 820–837, December 2004.

[7.] S. Yan, M. Mizumoto, and W. Z. Qiao, "An improvement to Kóczy and Hirota's interpolative reasoning in sparse fuzzy rule bases," Int. J. of Approximate Reasoning, vol. 15, pp. 185–201, 1996.

[8.] Z. H. Huang and Q. Shen, "Fuzzy interpolation with generalized representative values," in Proc. of the UK Workshop on Computational Intelligence, Loughborough, UK, September, 2004, pp. 161–171.

[9.] S. Jenei, "Interpolation and extrapolation of fuzzy quantities revisited - (I) An axiomatic approach," Soft Computing, vol. 5, pp. 179–193, 2001.

[10.] Johanyák Zs. Cs., Tikk D., Kovács Sz., és Wong K. W., Fuzzy rule interpolation Matlab toolbox – FRI toolbox, in Proc. of the IEEE World Congress on Computational Intelligence (WCCI'06), 15th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'06).

[11.] Z. Johanyák. Fuzzy rule interpolation matlab toolbox website. [Online]. Available: http://fri.gamf.hu

[12.] D. Tikk, Z. C. Johanyák, S. Kovács, and K. W. Wong, "Fuzzy rule interpolation and extrapolation techniques: Criteria and evaluation guidelines," Journal of Advanced Computational Intelligence and Intelligent Informatics,vol. 15, pp. 254–263, 2011.

[13.] Freemat website. [Online]. Available: http://freemat.sourceforge.net