JOHN ATANASOFF SOCIETY OF AUTOMATICS AND INFORMATICS

International Conference

AUTOMATICS AND INFORMATICS'12

PROCEEDINGS

Published by

JOHN ATANASOFF SOCIETY OF AUTOMATICS AND INFORMATICS

Bulgaria, Sofia, October, 3 – 5, 2012

JOHN ATANASOFF SOCIETY OF AUTOMATICS AND INFORMATICS

Secretariat Address
Bulgaria
1000 Sofia
108 Rakovsky str.

tel. (+359 2) 987 61 69
fax (+359 2) 987 61 69

e-mail: sai@infotel.bg
www.sai.infotel.bg
www.sai.bg

PROCEEDINGS CD: ISSN 1313-1869

ИЗПОЛЗВАНЕ НА РАЗМИТА ЛОГИКА И МЕТОДА НА ПРЕЦЕДЕНТИТЕ ЗА ПРЕДСКАЗВАЩА ДИАГНОСТИКА

IMPLEMENTATION OF FUZZY LOGIG IN CBR FOR FORECASTING DIAGNOSIS

Н. Христова, Л. Антонов

Химикотехнологичен и металургичен университет-София, Бул. Св.Климент Охридски 8, 1756 София, Тел (+3592) 8163134, E-mail: nchrist@uctm.edu

Abstract: Modern on-line process monitoring system should support classic fault detection, isolation and diagnosis sub-systems to avoid down-time, increase production, optimize parameters of the production line, etc. However faults usually demand immediate intervention by operator, therefore by using reliable prognostic system, risk can be avoided, maintenance intervals can be scheduled, operation and production strategy can be updated, etc. Case-Based Reasoning (CBR) is mostly used in designing the real time application having the decision support capability. The ability of CBR systems to apply cases to novel situations depends on their case adaptation knowledge. This paper describes implementation of fuzzy logic in the CBR systems that deriving effective knowledge representation schemes. It shows how this approach provides a framework for acquiring flexible adaptation knowledge from experiences with autonomous adaptation and suggests its potential as a basis for acquisition of adaptation knowledge.

Key words: Fuzzy Logic, Diagnosis, Maintenance, Case Based Reasoning (CBR)

ВЪВЕДЕНИЕ

При съвременните условия в индустрията се поставя задачата за създаване на цялостна система за предсказващо поддържане и нейното интегриране с оперативното и стратегическо ниво на общата система за мениджмънт на предприятието. Това налага използването на интелигентни техники за реализация на такава система [2, 3, 4, 5]. Разработването на системи за мониторинг и диагностика на машините и съоръженията в индустрията е област от съществено значение, осигуряваща значително намаляване на разходите за експлоатация, поддържане и ремонт, а също така и подобряване на мениджмънта, надеждността и безопасността. От особено значение е откриването и/или диагностицирането на потенциални проблеми и осъществяването на подходящи мерки за предотвратяването им, за да се избегнат сериозни последствия като аварии и прекъсвания на работата.

В настоящото изследване е представено комбиниране на метода на прецедентите с размита логика. Методът на прецедентите (Case-Based Reasoning – CBR) е широко използвана техника в системите за вземане на решение [1, 6, 7, 8], посредством която се намират решения за нови проблеми, базирайки се на предишни случаи (прецеденти) с известни решения.

ПОСТАНОВКА НА ЗАДАЧАТА – ОСНОВИ НА CBR

Класическият R^4 цикъл на CBR [1, 7] съдържа следните четири фази:

- Фазата *retrieval* извлича от базата-прецеденти най-близките до новия прецедент съхранени прецеденти. За тази цел се използват схеми за индексация и метрики за близост (подобие) [1, 6].
- *Reusing* – директно използване или интегриране на решенията на извлечените от предишната фаза прецеденти. Ако е необходимо решенията им се адаптират, за да се предложи решение, съобразено с формулирания целеви проблем.
- *Revising* – проверка на коректността и полезността на решението, генерирано на предишния етап.
- *Retaining* – запазване на новото решение в базата прецеденти като нов прецедент за бъдещо използване, ако новото придобито знание е значимо.

Основните предимства на метода на прецедентите са:

- осигурява гъвкавост при моделиране на знанието
- способност за изразяване на специфицирано знание
- естеството на представяне и описание на проблемите и решенията с прецеденти
- детерминираност и автономност на прецедентите
- лесно придобиване на знание – редуциране на задачата за придобиване на знание
- възможност за боравене с непълни и неточни данни и концепции
- приложение към проблеми, които са непълно дефинирани, при непредвидени (случайни) или липсващи стойности за входовете
- избягване повтарянето на минали грешки
- способност за анализ и правене на изводи
- разширяване към много различни цели и към по-широк обхват на областите на приложение.

Съществуват и някои недостатъци на CBR системите като: неспособност за изразяване на обобщено знание, трудности при извличане на знание при липса или ограничен (недостатъчен) обем от данни за разглежданата област; проблеми при извличането и адаптацията на прецедентите, осигуряване на обяснения за всички стъпки в процедурата.

Тези слабости и проблеми биха могли да се преодолеят с интегрирането на интелигентни техники и процедури в CBR системите. В тази разработка ще бъде представено комбинирането използване на размита логика и метода на прецедентита.

ПРИЛОЖЕНИЕ НА РАЗМИТА ЛОГИКА В CBR СИСТЕМИТЕ

В контекста на CBR размита логика (FL) [2, 4] се използва в етапите *индексиране* и *извличане* (търсене) на прецеденти. Размитата логика се използва при определяне на мярката за близост (подобие) на прецедентите. Размита логика (FL) (продукционни правила) се използва при адаптацията на прецедентите [3, 5, 9, 10]. Напр. извличане на размити продукционни правила при изследване на библиотеката от прецеденти и определяне на подобието между формулирания проблем и характеристиките на решенията за прецедентите.

Представяне на знанията от/за прецедент на базата на размитите множества. Едно размито множество ***A*** представлява съвкупност от обекти, която е подмножество на общото множество ***U***, с дефинирани функции на принадлежност, където на всеки обект ***x*** (***x*** ∈ ***U***) се съпоставя степен на принадлежност към лингвистичната променлива ***A*** [2, 4, 9]. Това формално се записва по следния начин:

$$A = \{\mu_A(x)/x, x \in U\}\ , \qquad (1)$$

където μ е функция на принадлежност и се дефинира с израза:

$$\mu_A : U \to [0,1]\ . \qquad (2)$$

Характеристиките на даден прецедент могат да бъдат представени с лингвистични променливи и съответните функции на принадлежност към тях. Процедурата за размиване представлява съпоставяне на детерминирани потребителски величини на принадлежността им към дефинираните лингвистични променливи. Размитата CBR система се състои от размита функция на подобие за сравняване на прецедентите в базата прецеденти и входния такъв. Уравнение за една камбановидна функция на принадлежност:

$$dist(c1,c2) = \sum_{k=1}^{n} w(k) * \left(1 - \frac{1}{1 + \left(\frac{|c1(k) - c2(k)|}{a}\right)^{2b}}\right) \qquad (3)$$

При разработването на CBR система често се налага да се определят някои свойства от базата прецеденти, за да се разделят прецедентите на подмножества. Реално обаче понякога е невъзможно да се дефинират класифициращи понятия в точна (детерминирана) форма. Например, при даден нов прецедент, може да не е възможно да се знае към кой клас принадлежи. Най-доброто знание, получено от миналите прецеденти, може само да ни даде достатъчно информация да се каже, че този прецедент принадлежи към границата между определени прецеденти, които могат да съдържат в себе си различни решения. Формулирането на тези долни и горни апроксимационни множества може да бъде обобщено до някакво случайно ниво на прецизност, което води до основите на идеята за точните апроксимации.

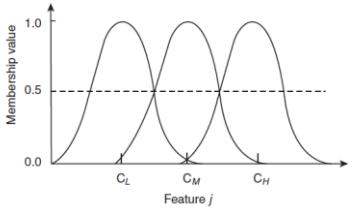
Лингвистично (размито) представяне на прецедентите. Тук се описва начинът на получаване на грануларно (зърнесто) пространство на характеристиките чрез използването на размито лингвистично представяне на прецедентите. Разглеждат се само числени характеристики, но тази структура може да работи и при характеристики с описателна форма или под формата на множество. Нека един обект $\hat{e}$ се представя с помощта на ***n*** числени характеристики (атрибути) $\hat{e} = [F_1, F_2, ..., F_n]$. Всеки атрибут е представен в границите на стойностите му на принадлежност, съответстващи на трите размити множества: ниско (low – ***L***), средно(medium – ***M***) и високо(high – ***H***). По този начин този ***n***-мерен вектор на обекта се представя в ***3n***-мерен вектор:

$$\hat{e} = [\mu_{low(F_1)}(\hat{e}), \mu_{medium(F_1)}(\hat{e}), \mu_{high(F_1)}(\hat{e}), \mu_{low(F_2)}(\hat{e}), \mu_{medium(F_2)}(\hat{e}), \mu_{high(F_2)}(\hat{e}), \ldots, \mu_{low(F_n)}(\hat{e}), \mu_{medium(F_n)}(\hat{e}), \mu_{high(F_n)}(\hat{e})] \qquad (4)$$

където $\mu_{low(F_j)}(\cdot), \mu_{medium(F_j)}(\cdot),$ and $\mu_{high(F_j)}(\cdot)$ показват степента на принадлежност на (.) към трите размити множества. За всеки входен атрибут F_j размитите множества ниско, средно и високо се характеризират индивидуално с π-функции на принадлежност:

$$\mu(F_j) = \pi(F_j, c, \lambda) = \begin{cases} 2\left(1 - \frac{||F_j - c||}{\lambda}\right) & \text{for } \frac{\lambda}{2} \le ||F_j - c|| \le \lambda \\ 1 - 2\left(\frac{||F_j - c||}{\lambda}\right)^2 & \text{for } 0 \le ||F_j - c|| \le \frac{\lambda}{2} \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

С λ>0 радиус на функцията и център ***c***. За всяко размито множество, λ и ***c*** имат различни стойности, които се избират така, че тези функции на принадлежност за трите множества да са припокриващи се при стойност 0.5 (Фиг.1).



Фиг. 1. Функции на принадлежност

Процедурата за избор на радиус и център на припокриващите се функции е следната: Нека m_j е средната стойност на всички точки около оста ***j***. Тогава m_{jl} и m_{jh} ще бъдат средните стойности на всички точки, съответно имащи координати в радиусите (F_{jmin},m_j) и (m_j,F_{jmax}). F_{jmin} и F_{jmax} означават долната и горната граници на динамичния радиус на атрибута F_j. Тогава центърът и радиусът се дефинират по следния начин:

$$\begin{aligned} c_{low(F_j)} &= m_{jl} \\ c_{medium(F_j)} &= m_j \\ c_{high(F_j)} &= m_{jh} \\ \lambda_{low(F_j)} &= c_{medium(F_j)} - c_{low(F_j)} \\ \lambda_{high(F_j)} &= c_{high(F_j)} - c_{medium(F_j)} \\ \lambda_{medium(F_j)} &= 0.5(c_{high(F_j)} - c_{low(F_j)}) \end{aligned} \qquad (6)$$

Генериране на правила на зависимост. Принципната задача е изчисляване на редуктори, отнасящи се за специален тип информационна система (такава за решения). Основните инструменти за това изчисляване са т.нар. ***d***-редуктори и ***d***-матрици на различимост.

Нека***Ts=[U,A]***е таблица с решения; ***Tc*** и ***Td*** =$\{d_1^*, d_2^*, ..., d_l^*\}$ са множества от атрибути на условие и решение. Тогава таблицата ***Ts=[U,A]*** се разделя на таблици $Ts_i=[U_i,A_i]$, съответстващи на ***l***-те атрибути на решението $d_1^*, d_2^*, ..., d_l^*$; $U = U_l \cup ... \cup U_l$ и $A_i = Tc \cup \{Td_i\}$. Ако $\{x_{i1}, x_{i2}, ..., x_{ip}\}$ е множество от тези обекти на U_i, които възникват в Ts_i, ***i=1,2,…..l,*** за всеки редуктор, нека ***B={b1,b2,…bk}***. Матрицата на различимост $M_{di}(B)$ от d_i-матрицата се дефинира:

$$c_{ij} = \{\text{Attr} \in B : \text{Attr}(x_i) \neq \text{Attr}(x_j)\} \qquad \text{for } i, j = 1, 2, \ldots, n \qquad (7)$$

За всеки обект $x_j \subset \{x_{i1}, x_{i2}, ..., x_{ip}\}$ се дефинира функция на различимост

$$f_{d_i^*}^{x_j} = \wedge\{\vee(c_{ij}) : 1 \le i,j \le n, j < n, c_{ij} \neq \emptyset\} \quad (8)$$

където всяко c_{ij} е разделяне (прекъсване на връзките) на всички членове от c_{ij}. Тогава функцията се конвертира в нормалната си разделителна форма. Правилата на зависимост r_i се получават по следния начин:

$$[\text{i.e., } d_i^* \leftarrow g_i, \quad (9)$$

където g_i е нормалната разделителна форма на $f_{d_i^*}^{xj}, j \in \{i_1, i_2, \ldots, i_p\}]$. Факторът на зависимост df_i за правилото r_i се определя с израза

$$\mathrm{df}_i = \frac{\mathrm{card}(\mathrm{POS}_i(d_i^*))}{\mathrm{card}(U_i)} \quad (10)$$

където $\mathrm{POS}_i(d_i^*) = \cup_{X \in I_{d_i^*}} l_i(X), l_i(X)$ е долната апроксимация на X с отношение към $I_{d_i^*}$. За реални данни, таблиците с решения често са несъвместими (непоследователни) и не могат да се получат перфектни/точни правила за решение. В такъв случай степента на точност на едно неточно правило се определя количествено от фактора на зависимост.

Генериране на прецеденти. Методологията за генериране на прецеденти в размитото гранулно пространство се състои от два етапа: генериране на размити правила с помощта на теорията на точните множества и напасването на правилата към прецедентите.

Напасване на правилата към прецедентите. Извършва се на базата на наблюдението, че всяко правило на зависимост (с честота над граничната) представлява клъстер в пространството на атрибутите. Може да се отбележи, че само подмножество от атрибути се появява във всяко от правилата, което показва, че не винаги е нужно цялото множество от атрибути да характеризира клъстер. Съставя се прецедент от правило на зависимост по следния начин:

1. Разглежда се лявата част от правилото. Тя се дели на елементарни правила, които съдържат само буквални връзки.

2. За всяко елементарно правило се генерира прецедент, съдържащ центровете и радиусите на присъстващите в правилото размити лингвистични променливи (от 1 правило могат да се получат много прецеденти).

3. Свързват се с всеки прецеден, генериран от предходната част от правилото и сила на прецедента, равна на фактора на зависимост на правилото. Силата на този фактор оказва влияние върху големината на съответния клъстер и значимостта на прецедента.

Така се получава прецедент със следната структура:

```
case{
Feature i: fuzzset_i: center, radius;
.....
Class k
Strength
}
```

Пример – разглеждат се данни с 2 атрибута F_1 и F_2 и два класа. От редуцираната таблица на атрибутите имаме две получени правила

$$\mathrm{class}_1 \leftarrow L_1 \wedge H_2, \mathrm{df} = 0.5,$$
$$\mathrm{class}_2 \leftarrow H_1 \wedge L_2, \mathrm{df} = 0.4,$$

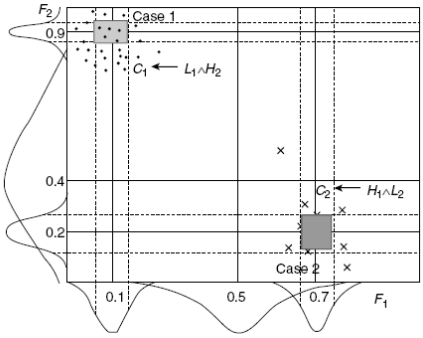
където L_1, L_2, H_1, H_2 означават съответно F_1 е ниско, F_2 е ниско, F_1 е високо, F_2 е високо; df – фактор на зависимост. Ако имаме следните параметри на размитите лингвистични множества ниско, средно и високо

Feature 1 : $c_L = 0.1, \lambda_L = 0.5, c_M = 0.5, \lambda_M = 0.7, c_H = 0.7, \lambda_H = 0.4.$

Feature 2 : $c_L = 0.2, \lambda_L = 0.5, c_M = 0.4, \lambda_M = 0.7, c_H = 0.9, \lambda_H = 0.5.$

Фиг. 2 илюстрира разгледания пример. Ще имаме и следните два класа

```
case 1 {
Feature No: 1, fuzzset (L): center=0.1, radius=0.5
Feature No: 2, fuzzset (H): center=0.9, radius=0.5
Class=1
Strength=0.5
}
case 2 {
Feature No: 1, fuzzset (H): center=0.7, radius=0.4
Feature No: 2, fuzzset (L): center=0.2, radius=0.5
Class=2
Strength=0.4
}
```



Фиг. 2. Извличане на размити правила и генериране на прецеденти

Извличане на прецеденти. Всеки получен на предния етап прецедент е набор от размити множества, описани от набор едномерни π-функции на принадлежност с различни стойности за център и радиус. За изчисляване на подобието на неизвестен n-мерен прецедент $\hat{e}$ спрямо e_p (с размерност n) се използва

$$\mathrm{SM}(\hat{e}, e_p) = \sqrt{\frac{1}{n_p}\sum_{j=1}^{n_p}[\mu_{fuzzset}^{j}(F_j)]^2}, \quad (11)$$

където $\mu_{fuzzset}^{j}(F_j)$ е степента на принадлежност на j-я компонент $\hat{e}$ към размитостта, представяща прецедента e_p. Когато $\mu^j = 1$ за всяко j и респективно $\mathrm{SM}(\hat{e}, e_p) = 1$, уравнение (11) представлява събирателна мярка, която се изчислява спрямо степента на подобие на всеки компонент на новия прецедент със съответстващия му такъв в запаметения прецедент. Размитите функции на принадлежност в (11) отчитат разпределението на точките вътре в гранулата, като по този начин осигуряват по-добро измерване на подобието, отколкото Евклидовото разстояние между две точки. За класификацията на новия прецедент, най-близкият такъв на базата на подобието се извлича и неговия класов знак се приписва на новия прецедент. Връзките между тях се разделят чрез параметъра "мощност на прецедента".

Концепция на размитите множества при измерване на подобието. Изводът на базата на размити правила за прецедентите (fuzzy case rule) може да се раздели на два етапа: на първия се прави извод на базата на това колко добре фактите на новия прецедент съответстват на елементите, свързани с прецедентното правило. За това се съди чрез използване на критерии ДА или НЕ, които се оценяват според степента на размита принадлежност между фактите и елементите. На втория етап се съставя извод за предишния в сравнение с новия прецедент, който се ръководи (насочва) от подобието между прецедентите. Заключенията, получени от двата етапа, се сравняват с тези от предишния прецедент. Ако заключенията са идентични с тези на предишния прецедент, то новият прецедент има същия краен резултат като предходния. Ако се разминават, решението, касаещо новия прецедент, не може да се потвърди от предходния.

Когато се прави преценка за съответствието между фактите на новия прецедент и елементите на прецедентното правило (то се представя от размити функции на принадлежност), не е необходима оценка ДА или НЕ за извод на прецедентното правило. Следователно, центърът на тежестта на размитите функции на принадлежност може да се дефинира с израза:

$$\mathrm{CG}(A_i) = \frac{\int_{c_1}^{c_2} x\mu_{A_i}(x)\,dx}{\int_{c_1}^{c_2} \mu_{A_i}(x)\,dx} \tag{12}$$

където Ai е размитото множество, което описва оценката на съответствие между елементите на прецедентното правило i и фактите на новия прецедент. Разстоянието между два центъра на тежестта се използва за описание на степента на подобие:

$$\mathrm{SM}(A,B) = 1 - |\mathrm{CG}(A) - \mathrm{CG}(B)| \tag{13}$$

за да задоволи условията на зависимостите на подобие.

Ако правилата се разгледат като обобщени, между два прецедента могат да се сравняват повече от 1 атрибут. Тогава се въвежда тегло в процеса на търсене и извличане, след което се претегля средната стойност на подобието.

$$\overline{\mathrm{SM}} = \frac{\sum (w_i \mathrm{SM}_i)}{\sum w_i} \tag{14}$$

Изчисляване на размито подобие между прецеденти. Съществуват няколко метода за изчисляване на подобието между прецеденти:

*) Числена комбинация от векторни характеристики (свойства, атрибути), представящи известният прецедент чрез използване на различна комбинация от правила;

*) Подобие при структурирано представяне – всеки прецедент е представен като структура (като насочен граф); тогава при измерването на подобие се вземат под внимание структурите на отделните атрибути на прецедента, а не само стойностите на атрибутите;

*) Водена от целта оценка на подобието – атрибутите на прецедента, с които ще се сравняват тези на новия такъв, зависят от желаната цел. С други думи, някои от атрибутите не са значими в основата на поставената цел и те не се вземат под внимание при изчисляването на степента на подобие;

*) Оценка на подобието на базата на правила – прецедентите в базата се използват за създаване на база от правила за вектора атрибути на всеки прецедент. Тази база от правила се използва за сравнение на прецедентите в базата прецеденти и за решаване на нов такъв;

*) Съвкупност от гореспоменатите методи според йерархията в областта на приложение.

Изграждането на размит прецедент и изчисляването на подобие може да се обобщи в преминаването през следните етапи:

1. Представяне на прецедента – за целта се използва вектор с три елемента. Елементите на този вектор описват свойството, неговата важност (тегло) вътре в прецедента и неговата стойност. Такава форма на представяне е обобщаване на често използваните други варианти, с допълнение, че има възможност всеки прецедент да бъде представен като отделно множество от свойства, чиито тегла са определени независимо.

$$e = \{\tau_0, \tau_1, \ldots, \tau_k\}$$
$$\tau_i = (P_i, w_i, \chi_i) \tag{15}$$

2. Подобие между размити и неразмити стойности на атрибутите.

Обобщена процедура за дефиниране на множество от оценки на подобие:

А) Създаване на междинен прецедент на базата на сравняваните такива – ако имаме два прецедента, описани с набор от свойства, се дефинира междинен прецедент, който има същия брой елементи, както $\cup$ (обединението) на двата прецедента.

$$e_q = \{\tau_0, \tau_1, \ldots, \tau_m\}$$
$$\Pi = P \cup P' \quad . \tag{16}$$

С $\boldsymbol{\tau_i}$ са означени междинните елементи (всеки от които е тройка "свойство, тегло и стойност"), които се използват по време на изчисляването на подобието. Множеството $\boldsymbol{\Pi}$ се състои от свойства, които описват междинния прецедент, а $\boldsymbol{m}$ е мощността на множеството $\boldsymbol{\Pi}$.

Б) Изчисляване на елементите (тройките) на междинния клас – в междинния клас се свързва свойство с всеки елемент $\boldsymbol{\tau_i}$ и се изчислява теглото и стойността за това свойство, като се използват тези от оригиналните прецеденти, които имат еднакви свойства

$$\tau_i = (\Pi_i, w_i, \sigma_i)$$
$$\Pi_i = P_j = P'_k$$
$$w_i = \mathrm{weight\_combination}(w_j, w'_k)$$
$$\sigma_i = \mathrm{similarity\_operator}(\chi_j, \chi'_k) \tag{17}$$

$\Pi i=Pj=P'k$ – едно от свойствата; w_i – комбиниран тегловен коефициент; σ_i – оператор на подобието между стойностите χj и $\chi'к$, който има стойност на подобието между 0 и 1. Тройката (P_j, w_j, χ_j) е елемент на прецедента $\boldsymbol{e}$, който има свойство $P_j=\Pi_j$.

В) Изчисляване на изходното (резултантното) измерване на подобието, използвайки междинния клас – след калкулирането на всеки от елементите на междинния клас, теглата се нормализират (сумата им да бъде 1). Тогава подобието между оригиналните прецеденти се изчислява от претеглената сума на дискретните стойности на подобието:

$$\overline{w_i} = \frac{w_i}{\sum_j w_j} \qquad \mathrm{SM}(e, e') = \sum_i \overline{w_i}\sigma_i \tag{18}$$

Тегловната комбинация е от съществена важност за качеството на оценката на подобието, както и операторът на подобие. Съществуват няколко метода за тегловна комбинация, които изменят теглата на стойностите на подобие за различните свойства на междинния прецедент, но не и стойностите на междинните прецеденти, както и самата процедура на сравнение.

В процеса на извличане и побдор на прецеденти в пространството на търсене е цялата база прецеденти. Това прави задачата не само трудна (скъпа) и неефективна, но също така понякога води до лошо представяне. За адресирането на такъв проблем се използват много алгоритми за класификация и клъстеризация преди избора на най-близките (подобни) един или повече прецеденти. След като прецедентът се раздели на няколко подклъстера, задачата за подбор и извличане се свежда до съответствие на новия прецедент с един или повече подклъстера и накрая може да бъде получен желаният брой подобни прецеденти.

За разработването на такъв алгоритъм за класификация/клъстеризация е важно да се предположи, че обектите могат да се представят в $\boldsymbol{n}$-мерно пространство, където осите представляват променливите (характеристиките), обектите (entity) стават точки (вектори), а клъстерите са компактна група от такива точки. Вътрешноточковите разстояния са зависими от използваната скала за измерване на променливите, както и от избраната метрика. Затова идеята на Евклидовото

пространство е важна при изграждането на метрики за подобие. Нещо повече, при реални проблеми се оказва, че клъстерите на прецедентите обикновено се припокриват, което значи, че те имат размити граници. Съществуват няколко метода за класификация и клъстеризация с размита логика – *Fuzzy ID3, Fuzzy c-Means*, като *Fuzzy ID3* е разширение на класическия такъв алгоритъми.

Интегриране на подходи за адаптация на базата на правила и на базата на прецеденти. Това е хибриден метод за адаптация на прецеденти, който използва комбинация от подходи на базата на правила и на прецеденти. Когато CBR системата срещне нов прецедент, най-близките до него като формулировка на проблема се извличат от базата прецеденти. Процесът на адаптация започва с малък набор от абстракни трансформационни правила. Тогава системата претърсва библиотеката с правила и намира нужната за изпълнение на трансформационните правила (за активиране на правилата) и ги прилага директно към проблемите. Системата подобрява адаптационните си способности чрез метода на базата на прецеденти, като го прилага в самия процес на адаптация на новия прецедент: предложеното решение се запаметява и се прилага повторно, ако в бъдеще се появят подобни проблеми. Ако новото решение все още не е достатъчно приемливо след прилагането на този комбиниран подход, потребителите могат да извършат специфична адаптация с помощта на потребителския интерфейс на системата.

Системата се състои от следните компоненти за адаптация:

- Базиран на правила (трансформиращи правила) компонент на адаптация – системата използва адаптационни правила, които след това се прилагат директно към проблема. Тези правила са получени чрез интервюиране на експертите в областта на проблема.
- Компоненти на адаптация на базата на прецеденти – Ако нито едно от правилата не е подходящо за приложение, системата опитва да извлече от базата прецеденти такъв за адаптация, който да описва успешна предишна адаптация на подобен като формулировка проблем. Ако се намери такъв прецедент, той се използва за адаптация на решението; в противен случай се започва процес на ръчна адаптация.
- Компонент за ръчна адаптация – Ако и двата по-горе описани процеси не успеят да дадат приемливо решение, потребителят може да извърши ръчна адаптация, използвайки потребителския интерфейс на системата.

При успешна адаптация на решението, то и стъпките за достигането до него се запаметяват за по-нататъшно ползване.

Чрез използване на адаптационна матрица. За проблеми, при които има взаимодействие между атрибутите (между два прецедента да има зависимост), се прилага т.нар. адаптационна матрица, която описва тази характеристика. Идеята на метода се описва в следната последователност: Атрибутите на прецедента се разделят/класифицират на две групи – независими (базови) и зависими (извлечени, произлизащи). Например, зависим атрибут може да бъде цената на някакъв продукт, а независими – необходимите за производството му компоненти и елементи. Тогава може да се разработи адаптационна матрица, която описва връзките (или нивата на влияние) на независимите атрибути върху зависимите такива. С помощта на тази матрица може да се изчисли ефектът върху зависимите атрибути при промяна на независимите. При поява на нов прецедент, системата извлича от базата прецеденти всички максимално подобни. Разликите между текущия прецедент и извлечените от базата се отбелязват в тази адаптационна матрица, след което може да се направи адаптация на извлечените решения. Например: ако се променят независимите параметри, които имат минимален ефект върху зависимите такива, е необходима само ограничена адаптация, респективно и за решенията. Ако се променят независимите атрибути, които обаче имат съществен ефект върху зависимите, предложените решения ще имат нужда от дълга и голяма настройка.

Размито дърво на решението. Размитото дърво на решение освен за класификация на прецедентите, може да се използва и за представяне на знания за адаптация на решение. В следващите няколко стъпки е показан начинът, по който се добиват подобни знания:

1. Запомнят се теглата на характеристиките на атрибутите на прецедентите.
2. Прецедентите се клъстеризират, като за целта се използва претеглена метрика.
3. За всеки прецедент във всеки клъстер се сравняват разликите между атрибутите на прецедентите и на решението за този прецедент и останалите. Така се получава вектор на несъответствието.
4. Размива се полученият вектор на несъответствието.
5. За всеки един прецедент се получава едно размито дърво на решението чрез обучение с използването на размития вектор на несъответствието. Генерират се размити правила за решение от дървото и правилата се опростяват с помощта на метода на детерминираните множества.
6. При поява на нов проблем се извлича най-подобният прецедент от базата, след което се размива разликата в описанието на прецедента между тези два прецеденти. Получената размита разлика се използва като вход на размитото дърво на решение. Получават се настройки под формата на лингвистични променливи с фактори на сигурност. Те се деразмиват с цел получаване на детерминирана настройка.
7. Използват се получените стойности за настройка за адаптация на старото решение.

В стъпка 1 теглата на характеристиките на атрибутите могат да се получат с използването на някои от методите, представени по-горе. За стъпка 2, клъстеризацията може да се направи по метода на ***к***-тия най-близък съсед или чрез **fuzzy c-means**. Деразмиването може да се извърщи с прилагането на някои от методите за деразмиване.

Методологията за поддръжка на базата прецеденти (Case Base Maintenance) съдържа следните етапи:

1. Целта на тази фаза е да се научат/запомнят теглата на характеристиките на всеки прецедент. Това се прави с помощта на оценяваща функция, която не е нужно да има информация за класа. Използвайки информацията на теглата, оригиналното пространство на характеристиките се трансформира в такова на претеглените характеристики, така че клъстеризацията на прецедентите може да бъде по-ефективна.
2. Целта тук е да се раздели базата с прецеденти на няколко клъстера чрез претеглена метрика на разстоянието, базирана на концептите за вътре-клъстерното и междуклъстерните подобия. Във всеки от тези клъстери се идентифицират няколко представителни прецедента, а непредставителните прецеденти в клъстера се апроксимират чрез тях. Тази апроксимация се извършва с помощта на група от адаптационни правила.

3. Тук се цели добиването на тези адаптационни правила на базата на размит подход. Детерминираните множества се използват за премахването на ненужните атрибути чрез редукторите и по този начин се редуцира времето за търсене. Размитите множества се използват за описание на апроксимационните възможности на размитите адаптационни правила и тези от тях, които имат силно изразена апроксимационните способност, след това се избират като представителни, като по този начин се намалява времето за търсене. Такива представителни правила се считат (разглеждат) като адаптационно знание, свързано с представителните прецеденти.

4. На базата на правилата за адаптация, на този етап се използва механизъм на логически извод за предсказване настройката на решението за прецедента-заявка.

5. Тази стъпка описва как се избират няколко представителни прецеденти от всеки клъстер на базата на концепцията за обхвата и достъпността. Тъй като множествата, представящи обхвата и достъпността на различни прецеденти са неточно дефинирани, понятията за размити множества са обединени. Така получените представителни размити правила за адаптация и прецеденти представляват редуцираната част на потребителя в разпределената CBR система.

6. Тук се извършват периодичното обновяване (актуализиране) и повторният избор на представителните прецеденти, поради промяна в полученото знание и природата на задачите. Тази поддръжка се извършва чрез повторно прилагане на описаните в предходните 5 етапа алгоритми. Важно е да се отбележи, че се контролира/определя броят на представителните прецеденти в клъстера и по този начин се контролира и големината на базата прецеденти.

ЗАКЛЮЧЕНИЕ

В заключение, ползите от използването на размита логика могат да се обобщят по следния начин:

- Числовите характеристики се представят с лингвистични променливи (функции на принадлежност) за по-лесно сравнение;
- Размитите множества позволяват многократно индексиране (multiple indexing) за един прецедент с различни степени на принадлежност;
- Размитата логика улеснява трансфера на знание за различни области;
- Размитите множества, използвайки изменящи се показатели, повишават гъвкавостта на процеса на извличане на прецеденти.

Бъдещите изследвания ще са насочени към създаване на диагностични модели, които ще бъдат използвани за целите на предсказващата диагностика и поддръжка на съоръженията.

БЛАГОДАРНОСТ

Изследванията са финансирани от Фонд "Научни изследвания" (Ф"НИ") към Министерството на образованието, младежта и науката по проект „Предсказващо поддържане на технологични съоръжения въз основа на диагностика и анализ на риска" № ДВУ-10-0267/10.

ЛИТЕРАТУРА

1. Aamodt, A. and Plaza, E. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, *Artificial Intelligence Communications*, vol. 7, no. 1, pp 39-59, 1994.

2. Amit Konar, *Computational Intelligence: Principles, Techniques and Applications*, Springer, New York, 2005.

3. Chan, F. T. S. Application of a Hybrid Case-Based Reasoning Approach in Electroplating Industry, *Expert Systems with Applications*, 29(1), pp. 121-130, 2005.

4. Fakhreddine, O. Karray & Clarence De Silva, Soft Computing and Intelligent Systems Design, *Pearson Education*, pp. 126-128, 2009.

5. Hanemann, L. A Hybrid Rule-Based/Case-Based Reasoning Approach for Service Fault Diagnosis, in *Proceedings of the 20th IEEE International Conference on Advanced Information Networking and Applications*, Vienna, Austria, April, 2006.

6. Jakobson, G., J. Buford, and L. Lewis, Towards an Architecture for Reasoning about Complex Event-Based Dynamic Situations", in *Proceedings of the 26th International Conference on Software Engineering*, May 2004, IEE.

7. Kolodner, J. *Case-Based Reasoning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

8. Mitra, R., J. Basak, Methods of case adaptation: A survey, *International Journal of Intelligent Systems*, 20(6), pp. 627-645, 2005.

9. Pal, S.K, and P. Mitra, Case Generation using Rough Sets with Fuzzy Representation, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, March 2004.

10. Sarkar, S. and S. Karforma, An Assessment Evaluator Using Fuzzy Distance Approach, *International Journal of Computer Engineering and Information Technology*, Spring Edition, 2009.