LMD mokslo darbai, III tomas, 230–235

Kai kurie teisinių žinių bazių lyginamosios analizės aspektai

Laima PALIULIONIENĖ (MII)
el. paštas: laipal@ktl.mii.lt

1. Įvadas

Vienas iš procesų, kuris galėtų būti iš dalies automatizuotas kompiuterinėje sistemoje, skirtoje padėti teisinius dokumentus rengiančioms grupėms, yra teisinių dokumentų lyginamoji analizė. Lyginamoji analizė yra svarbi dviem pagrindiniais atvejais: a) kai reikia palyginti skirtingus ruošiamo teisinio dokumento arba jo dalies variantus, siekiant pasirinkti geriausią, b) kai reikia palyginti skirtingų valstybių analogiškus įstatymus.

Šiame straipsnyje palyginimą mes suprantame kaip procesą, kurio metu nustatomos situacijos, turinčios skirtingas teisines pasekmes pagal skirtingus įstatymus arba jų variantus. Intelektualizuotoje kompiuterinėje sistemoje teisiniai dokumentai formalizuotu pavidalu saugomi žinių bazėse. Tam mes naudojame freimų logikos (F-logikos) kalbą. Taigi, teisinių dokumentų palyginimo uždavinys suvedamas į žinių bazių palyginimo uždavinį. Pažymėsime, kad žinių bazių palyginimas nėra pakankamai išnagrinėtas ne vien teisinių žinių bazių srityje, bet ir apskritai žinių bazių teorijoje. Todėl pirmiausia reikia ištirti, kokiais būdais galima ieškoti skirtumų tarp kelių mus dominančio tipo žinių bazių.

Ankstesniuose darbuose [1, 2] mes pasiūlėme ir teoriškai pagrindėme algoritmą, skirtą lyginti žinių bazes, išreikštas Horno taisyklių loginėmis programomis. Šiame straipsnyje nagrinėjamas bendresnis atvejis – kai žinių bazės išreikštos bendrosiomis loginėmis programomis. Pateikiamas algoritmas skirtumams tarp žinių bazių nustatyti, kuris naudoja tam tikru būdu sukonstruotas testines situacijas.

2. Teisinių dokumentų vaizdavimas

Teisinių dokumentų vaizdavimo formalizmo pagrindu mes siūlome naudoti freimų logiką (F-logiką). F-logika [3, 4] – tai formalizmas, jungiantis objektinį žinių vaizdavimo būdą su loginio išvedimo mechanizmu, grindžiamu rezoliucija. Žinių bazę sudaro taisyklės, kurių forma yra $C \leftarrow A$, kur C yra F-molekulė, A yra F-molekulių konjunkcija. F-molekulėse naudojami sąvokų vardai, atributų vardai, atributų reikšmės, predikatų simboliai. Sąvokoms, atributams ir jų reikšmėms žymėti naudojami pirmos eilės termai sudaryti iš funkcijų simbolių ir kintamųjų, kaip ir predikatų skaičiavime. Jie vadinami identifikaciniais termais (id-termais). Faktai yra taisyklės su tuščiu antecedentu.

1-asis pavyzdys. Teiginys „atsakomybė už pasikėsinimą atsiranda pagal tą baudžiamąjį įstatymą, kuris numato atitinkamą baigtą nusikaltimą" F-logikoje formalizuojamas taip:

atsako(N, P, S)

$\leftarrow P$: *pasikėsinimas*[*subjektas*$\rightarrow N$, *pradeda_ką*$\rightarrow V$],

S : *baudžiamasis įstatymas*[*už_ką*$\rightarrow V$], V : *nusikaltimas*.

Čia predikatas *atsako*(N, P, S) suprantamas kaip „N atsako už P pagal S".

Atributų reikšmės F-logikoje gali būti perduodamos pagal nutylėjimą. Tuo tarpu F-logikos dalis be tokio atributų reikšmių perdavimo yra ekvivalentiška predikatų skaičiavimui, t.y. tarp jų egzistuoja abipusis atvaizdavimas. Šiame straipsnyje nagrinėsime tik šią F-logikos dalį. Kad būtų paprasčiau, toliau pavyzdžiuose naudosime Prologo notaciją, o ten, kur nenaudojami kintamieji – teiginių simbolius vietoj predikatų su argumentais. Nagrinėsime bendrąsias programas, t.y. programas, kurias sudaro taisyklės $C \leftarrow L_1, \ldots, L_n (n \geqslant 0)$, kur C yra atomas, o $L_1, \ldots, L_n$ – literos (atomai arba jų neiginiai).

F-logikos išvedimo mechanizmas įgalina gauti atsakymą į užklausą panašiai kaip Prologe. Kad galėtume gauti visas konkrečios situacijos išvadas, mes išplečiame F-logiką formaliu įrodymu, grindžiamu modelių generavimu. Naudojama tokia modelio generavimo taisyklė, atspindinti neigimo kaip nesėkmės taisyklę:

Modelio generavimo taisyklė

Pradinis modelis yra tuščia aibė. Tai, kad iš žinių bazės KB galima sugeneruoti modelį N, žymėsime $KB \vdash_M N$. Modelis generuojamas taip:

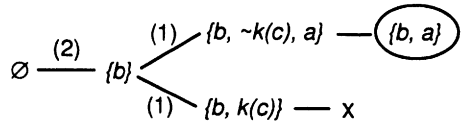
1. *Horno disjunktai.* Jei žinių bazėje yra taisyklė $C \leftarrow A$ ir egzistuoja keitinys σ toks, kad $A\sigma$ yra teisingas iki šiol sukonstruotame modelyje N, o $C\sigma$ šiam modeliui nepriklauso, tai N išplečiamas įtraukiant į jį $C\sigma$.

2. *Bendrieji disjunktai.* Jei žinių bazėje yra taisyklė $C \leftarrow A, notP_1, \ldots, notP_n (n > 0)$, kur A susideda iš atomų ir $P_1, \ldots, P_n$ yra atomai, ir egzistuoja keitinys σ toks, kad $A\sigma$ yra teisingas iki šiol sukonstruotame modelyje N, o $C\sigma$ šiam modeliui nepriklauso, tai N išplečiamas, gaunant du modelius su specialiu modaliniu operatoriumi „k" – modelį $N \cup \{C\sigma, \neg k(P_1\sigma), \ldots, \neg k(P_n\sigma)\}$ ir modelį $N \cup \{k(P_1\sigma; \ldots; P_n\sigma)\}$. Čia $\neg k(P)$ reiškia, kad tikimasi, jog modelis neturės fakto P, o $k(P_1; \ldots; P_n)$ reiškia, kad tikimasi, jog modelis turės bent vieną iš faktų $P_1, \ldots, P_n$. Toliau generuojant modelius atmetami tie, kuriuose yra $\neg k(P)$ ir P. Jei modelyje yra $k(P_1; \ldots; P_i; \ldots; P_n)$ ir P_i $(i = 1, \ldots, n)$, iš jo pašalinamas $k(P_1; \ldots; P_i; \ldots; P_n)$. Iki galo sugeneravus visus modelius atmetami tie, kuriuose yra $k(P_1; \ldots; P_n)$, bet nėra nė vieno P_i $(i = 1, \ldots, n,\ n > 0)$. Galutiniai modeliai gaunami iš neatmestų modelių pašalinus faktus su operatoriumi „k".

Pagal šią taisyklę į modelį pirmiausia įtraukiami faktai, kadangi faktai yra taisyklės su tuščiu antecedentu. Toliau pagal taisyklių grandines gaunamos išvados iš šių faktų. Skirtingai nuo Horno atvejo, modelis nebūtinai bus vienas – jų gali būti keli arba iš viso nebūti. Pademonstruosime modelio generavimą paprasčiausiu atveju, kai nėra kintamųjų.

2-asis pavyzdys. Ši žinių bazė turi vieną modelį $\{b, a\}$:

$$a \leftarrow b,\ not\ c. \qquad (1)$$



b. (2)

Jos modelis generuojamas taip (× reiškia modelio atmetimą):

3-asis pavyzdys. Ši žinių bazė turi du modelius – $\{a\}$ ir $\{c\}$.

$a \leftarrow not\ c.$

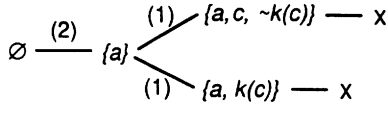
$c \leftarrow not\ a.$

4-asis pavyzdys. Ši žinių bazė neturi modelio:

$c \leftarrow a,\ not\ c.$ (1)

$a.$ (2)

Šios žinių bazės modelis generuojamas taip:



3. Skirtumų tarp žinių bazių analizė

Nagrinėjamas žinių bazes ir situacijas apribosime šiomis sąlygomis:

Surištų kintamųjų sąlyga. Visi kintamieji taisyklių konsekventuose ir antecedentų neigiamose dalyse yra surišti, t.y. naudojami teigiamoje antecedento dalyje.

Vienintelio modelio sąlyga. Generuojamas modelis yra tiksliai vienas (ši sąlyga tinka teisės dalykinėje srityje).

Baigtinio modelio sąlyga. Generuojamas modelis yra baigtinis.

Apibrėžkime, ką mes vadinsime skirtumu tarp teisinių žinių bazių, kai jas lyginsime.

1-oji apibrėžtis. Tarp žinių bazių KB_1 ir KB_2 yra *vidinis skirtumas*, jei $KB_1 \vdash_M R_1$, $KB_2 \vdash_M R_2$, ir rezultatų aibės R_1 ir R_2 nesutampa. Žymėsime taip:

$DiffInt(KB_1, KB_2) = (R_1 \cup R_2) \backslash (R_1 \cap R_2)$ – vidinio skirtumo aibė.

2-oji apibrėžtis. Tegu S yra faktų aibė, aprašanti situaciją. Žinių bazės KB_1 ir KB_2 yra *skirtingos situacijos S atžvilgiu*, jei $KB_1 \cup S \vdash_M R_1$, $KB_2 \cup S \vdash_M R_2$, ir rezultatų aibės R_1 ir R_2 nesutampa. Žymėsime taip:

$Diff(S, KB_1, KB_2) = (R_1 \cup R_2) \backslash \big((R_1 \cap R_2) \cup DiffInt(KB_1, KB_2)\big)$ – skirtumo aibė (t.y. į skirtumo aibę mes neįtraukiame faktų iš vidinio skirtumo aibės).

3-oji apibrėžtis. Žinių bazės yra *skirtingos*, jei tarp jų yra vidinis skirtumas arba jos yra skirtingos kurios nors situacijos atžvilgiu.

Rasti visus skirtumus tarp teisinių žinių bazių reiškia rasti vidinio skirtumo aibę, visas situacijas, kurių atžvilgiu žinių bazės yra skirtingos, ir šių situacijų skirtumo aibes. Galimų situacijų gali būti labai daug, o kadangi id-termuose gali būti vartojami funkcijų simboliai, tai galimų situacijų yra begalinė aibė. Tiesiogiai visų jų patikrinti neįmanoma. Parodysime, kad vietoj visų įmanomų situacijų tikrinimo pakanka patikrinti testines situacijas, sukonstruotas iš konkretizuotų taisyklių antecedentų.

4-oji apibrėžtis. Teigiamos (be neiginių) formulės *konkretizacija yra elementari*, jei keitinyje, kuriuo gauta ši konkretizacija, visi kintamieji keičiami į konstantas ir:

1) šiose konstantose nėra funkcijų simbolių,

2) konstantos nepriklauso nagrinėjamų žinių bazių Herbrand'o universumui, t.y. nenaudojamos žinių bazėse,

3) skirtingi kintamieji keičiami į skirtingas konstantas.

5-oji apibrėžtis. Tegu yra lyginamos dvi žinių bazės. Faktų aibė E yra *elementari testinė situacija*, gauta iš vienos žinių bazės taisyklės $C \leftarrow A, notP_1, \ldots, notP_n$ ($n \geqslant 0$, A susideda iš atomų arba yra tuščia aibė, $P_1, \ldots, P_n$ yra atomai), jei teisingas vienas iš šių teiginių:

1) $E = A\theta$, kur $A\theta$ yra formulės A elementari konkretizacija (jei $A = \emptyset$, ši situacija nenagrinėjama),

2) $A\theta \subset E$, kur $A\theta$ yra formulės A elementari konkretizacija, ir bent vienai kitos žinių bazės taisyklei $D \leftarrow B, notR_1, \ldots, notR_m$ ($m > 0$, B susideda iš atomų arba yra tuščia aibė, $R_1, \ldots, R_m$ yra atomai), kurioje $B\theta \backslash V \subseteq A\theta$, kur V yra vidinių KB_2 faktų aibė, egzistuoja R_i ($1 \leqslant i \leqslant m$) toks, kad $R_i\theta \in E$.

Siūlomas elementarios testinės situacijos sudarymo algoritmas

1) Sudaryti taisyklės teigiamos dalies elementarią konkretizaciją taip: kiekvieno kintamojo vardas (kuris kaip ir Prologe prasideda didžiąja raide) keičiamas į konstantos vardą, tiesiog keičiant didžiąsias raides į mažąsias. Jei gauta konstanta priklauso Herbrand'o universumui, prie jos vardo pridedamas ženklas „1" (pavyzdžiui, vietoj „abc" gauname „abc1"). Jei ir toks vardas priklauso HerbrandÆo universumui, tas pats ženklas pridedamas dar kartą ir tai daroma tol, kol gaunama norima konstanta. Šis procesas bus baigtinis, nes žinių bazės konstantų be funkcijų simbolių yra baigtinė aibė.

2) Jei antroje žinių bazėje yra taisyklės su neigiama dalimi, tenkinančios 5-osios apibrėžties antrą punktą, sukonstruoti papildomas elementarias testines situacijas, kuriose būtų įvairios kombinacijos faktų iš neigiamų tų taisyklių dalių (pagal 5-osios apibrėžties antrą punktą).

Pastebėkime, kad Horno taisyklių atveju šis algoritmas sutampa su ankstesniame darbe [2] pateiktu algoritmu – tuomet tiesiog nenaudojamas antras žingsnis.

5-asis pavyzdys. Žemiau pateiktos dvi žinių bazės, jų palyginimui sukonstruotos elementarios testinės situacijos ir iš jų sugeneruoti modeliai. Šios žinių bazės yra skirtingos situacijos $\{S(x),\ P(a)\}$ atžvilgiu.

Teorema 1. *Žinių bazės KB_1 ir KB_2, kurios išreikštos bendrosiomis programomis ir tenkina aukščiau išvardintas sąlygas, yra skirtingos tuomet ir tik tuomet, kai tarp šių žinių bazių yra*

KB_1
$R(X) \leftarrow S(X),\ not\ P(a)$ $Q(X) \leftarrow S(X),\ not\ R(X)$
$\{S(x)\} \cup KB_1 \vdash_M \{S(x), R(x)\}$ $\{S(x), P(a)\} \cup KB_1 \vdash_M \{S(x), P(a), Q(x)\}$ $\{S(x), R(x)\} \cup KB_1 \vdash_M \{S(x), R(x)\}$ $\{S(x), P(a), R(x)\} \cup KB_1 \vdash_M \{S(x), P(a), R(x)\}$

KB_2
$R(X) \leftarrow S(X)$
$\{S(x)\} \cup KB_2 \vdash_M \{S(x), R(x)\}$ $\{S(x), P(a)\} \cup KB_2 \vdash_M \{S(x), P(a), R(x)\}$ $\{S(x), R(x)\} \cup KB_2 \vdash_M \{S(x), R(x)\}$ $\{S(x), P(a), R(x)\} \cup KB_2 \vdash_M \{S(x), P(a), R(x)\}$

vidinis skirtumas arba egzistuoja elementari testinė situacija E, gauta iš vienos iš KB_1 arba KB_2 taisyklių pagal aukščiau siūlomą algoritmą, kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos.

Įrodymo idėja. Teoremoje suformuluotos sąlygos pakankamumas gaunamas tiesiogiai pagal 3-ąją apibrėžtį. Jei tarp žinių bazių yra vidinis skirtumas, teoremos sąlygos būtinumas taip pat seka iš 3-iosios apibrėžties. Kai vidinio skirtumo nėra, šios sąlygos būtinumas įrodomas prieštaros būdu. Iš tikrųjų, jei elementarių testinių situacijų atžvilgiu žinių bazės nėra skirtingos, tai skirtumas kurios nors kitos situacijos atžvilgiu galėtų atsirasti dėl to, kad vienos arba kelių taisyklių neigiamoje dalyje yra faktai, dėl kurių į modelį neįtraukiamas šios taisyklės konsekventas. Tačiau šios neigiamos dalys įtrauktos ir į atitinkamas elementarias testines situacijas, todėl būtų gautas ir skirtumas šių situacijų atžvilgiu. O tai prieštarauja mūsų prielaidai. Tai, kad testinėje situacijoje pakanka naudoti elementarias konkretizacijas, įrodoma analogiškai kaip ankstesniame darbe [2], t.y. analizuojant išvadas iš skirtingų keitinių variantų.

Remdamiesi šia teorema, pateiksime skirtumo tarp teisinių žinių bazių nustatymo algoritmą.

Skirtumo tarp teisinių žinių bazių nustatymo algoritmas

Įvestis. Žinių bazės KB_1 ir KB_2.

Išvestis. (1) Vidinio skirtumo aibė $DiffInt(KB_1, KB_2)$ ir taisyklių aibės, kurių pagalba gautas kiekvienas faktas iš vidinio skirtumo aibės. (2) Elementarių testinių situacijų, kurioms $Diff(E, KB_1, KB_2) \neq \emptyset$, aibė; kiekvienai tokiai situacijai E – skirtumo aibė $Diff(E, KB_1, KB_2)$, o kiekvienam faktui iš šios aibės – taisyklių aibės, kurių pagalba jis gautas.

1-asis žingsnis. Atliekamas vidinis modelio generavimas ir gaunama vidinio skirtumo aibė $DiffInt(KB_1, KB_2)$. Jei $DiffInt(KB_1, KB_2) \neq \emptyset$, kiekvienam $d \in DiffInt(KB_1, KB_2)$ gaunama tokia konkretizuotų taisyklių seka $T = [t_0, \ldots, t_n]$, kad: (a) kiekvienam i $(0 \leqslant i \leqslant n)$ ir kiekvienam taisyklės t_i teigiamos antecedento dalies elementui A_{ij} galioja, kad arba A_{ij} yra vidinis žinių bazės faktas, arba egzistuoja $k < j$ toks, kad A_{ij} yra t_k konsekventas, (b) kiekvienam i $(0 \leqslant i \leqslant n)$ ir kiekvienam taisyklės t_i neigiamos antecedento dalies antecedento elementui *not* P_{ij} galioja, kad P_{ij} nepriklauso KB_m modeliui ($m = 1$ arba $m = 2$, priklausomai nuo to, kurioje bazėje gaunams d) , (c) d yra t_n konsekventas.

2-asis žingsnis. Sudaroma testinių situacijų aibė. Ji sudaroma pagal aukščiau siūlomą algoritmą iš kiekvienos taisyklės iš KB_1 ir KB_2.

3-asis žingsnis. Kiekvienai testinei situacijai E ir kiekvienam faktui iš $Diff(E, KB_1, KB_2)$ gauname taisyklių sekas, kaip 1-ame žingsnyje (tik dalyje (a) atsiranda papildoma alternatyva, kad $A_{ij} \in S$).

4. Išvados

Straipsnyje parodyta, kad žinių bazių, išreikštų bendrosiomis loginėmis programomis, palyginimui pakanka tikrinti elementarias testines situacijas ir tai leidžia išvengti begalybės, atsirandančios dėl visų galimų situacijų testavimo. Taigi, teisinių dokumentų lyginamąją analizę galima iš dalies automatizuoti, formalizavus juos ir taikant straipsnyje siūlomą žinių bazių palyginimo metodą. Automatizuoto skirtumo nustatymo rezultatai po to turi būti pateikiami teisininko-eksperto analizei.

Literatūra

[1] L. Paliulionienė, Loginis išvedimas ir teisinių žinių bazių integravimas: pagrindiniai algoritmai, *Lietuvos matematikų draugijos mokslo darbai*, II tomas, Technika, Vilnius (1998), pp. 199–205.

[2] L. Paliulionienė, Požiūrių vaizdavimas ir jų skirtumų analizė teisinių dokumentų rengimo sistemų žinių bazėse, *Lietuvos mokslas ir pramonė, Informacinės technologijos-99, Konferencijos pranešimų medžiaga*, Technologija, Kaunas (1999). Įteikta spaudai.

[3] M. Kifer, G. Lausen, and J. Wu, *Logical Foundations of Object-Oriented and Frame-Based Languages*, Technical Report 93/06, Department of Computer Science, University SUNY at Stony Brook (1993).

[4] A. Čaplinskas, L. Paliulionienė, and S. Sinkevičiūtė, Project NEMESIS: Knowledge-based aspects in law engineering systems, in: *DEXA'95 Workshop Proceedings*, London (1995), pp. 413–421.

Some aspects of the comparative analysis of legal knowledge bases

L. Paliulionienė

One of the tasks in the process of preparing new legislation is the comparative analysis of different versions of the legislation. In this article we propose how the analysis could be partly automated if the legislation is formalized and stored in a knowledge base. The algorithm is proposed to find differences between legislations by using certain test situations. The algorithm is based on derivation by model generation and it is grounded in the case when the knowledge bases are general logic programs.