LMD mokslo darbai, 199–205
1998, Vilnius

Loginis išvedimas ir teisinių žinių bazių integravimas: pagrindiniai algoritmai

L. Paliulionienė (MII)

Įvadas

Rengiant teisinius dokumentus (įstatymus, nutarimus ir kt.), neišvengiamai iškyla skirtingų dokumentų arba jų fragmentų suderinimo tarpusavyje ir integravimo uždavinys. Intelektualizuotoje kompiuterinėje sistemoje, skirtoje padėti teisinių dokumentų rengėjams, šis uždavinys susijęs su žinių bazių analize ir integravimu. Teisinių žinių bazių integravimą galima nagrinėti keliais aspektais. Vienas aspektas yra teisinio dokumento skirtingų variantų lyginamoji analizė. Integravimas šiuo atveju reiškia labiausiai priimtino varianto, į kurį gali būti įtraukti teiginiai iš kelių variantų, pasirinkimas. Šį integravimo uždavinį sprendžia teisininkas-ekspertas, o sistemos paskirtis yra pateikti jam teisinio dokumento variantų lyginamosios analizės rezultatus. Tokia analizė yra svarbi ir lyginant skirtingų valstybių įstatymus toje pačioje teisinėje srityje. Kitas aspektas yra teisinių dokumentų integravimas, užtikrinant jų tarpusavio suderinamumą. Tai gali būti skirtingų autorių parengtų dokumento fragmentų integravimas, kurio rezultatas turi būti vienas suderintas dokumentas. Tai gali būti ir skirtingų teisinių dokumentų integravimas, kurio tikslas yra užtikrinti teisinių dokumentų sistemos suderinamumą. Tai ypač svarbu įvedant naują įstatymą arba kitokį teisinį dokumentą į teisinę sistemą.

Šiame straipsnyje nagrinėjami du pagrindiniai teisinių žinių bazių integravimo aspektai: žinių bazių palyginimas ir jų sujungimas, užtikrinant suderinamumą. Siūlomi pagrindiniai algoritmai tokiam integravimui atlikti.

Teisinių žinių bazių palyginimas

Kaip jau buvo pasakyta, teisinių dokumentų palyginimas reikalingas nagrinėjant kelis to paties dokumento variantus arba skirtingų valstybių įstatymus toje pačioje teisinėje srityje. Lyginimu mes vadinsime tokį procesą, kurio metu nustatomos situacijos, kurios turi 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. F-logika [1, 2] – tai formalizmas, jungiantis objektinį žinių vaizdavimo būdą su loginiu 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.

Atributų reikšmių perdavimas pagal nutylėjimą F-logikoje įgalina modeliuoti nemonotonines žinias. Tuo tarpu monotoninė F-logikos dalis ekvivalentiška predikatų skaičiavimui, t.y. tarp jų egzistuoja abipusis atvaizdavimas. Šiame straipsnyje mes nagrinėsime monotoninę F-logikos dalį.

1 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*–>*N, pradeda_ką*–>*V*],

S : *baudžiamasis_įstatymas*[*už_ką*–>*V*], *V* : *nusikaltimas*.

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

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

Modelio generavimo taisyklė. Jei žinių bazėje yra taisyklė $C \leftarrow A$ ir egzistuoja keitinys σ toks, kad $A\sigma$ yra teisingas iki šiol sukonstruotame modelyje *M*, o $C\sigma$ jame nėra teisingas, tai *M* praplečiamas įtraukiant į jį $C\sigma$. Pradinis modelis yra tuščia aibė. Tai, kad iš žinių bazės *KB* pagal šią taisyklę galima sugeneruoti modelį *M*, žymėsime $KB \vdash M$.

Pagal šią taisyklę į modelį pirmiausia įtraukiami faktai, o toliau pagal taisyklių grandines gaunamos išvados iš šių faktų.

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

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

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

DiffInt(KB_1, KB_2)KB_1 = $R_1 \setminus R_2$ – faktų, generuojamų iš KB_1, bet negeneruojamų iš KB_2 aibė,

DiffInt(KB_1, KB_2)KB_2 = $R_2 \setminus R_1$ – faktų, generuojamų iš KB_2, bet generuojamų iš KB_1 aibė.

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

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

DiffInt(*S*, KB_1, KB_2)KB_1 ir *DiffInt*(S, KB_1, KB_2)KB_2 – analogiškai kaip 1-oje apibrėžtyje.

Rasti visus skirtumus tarp teisinių žinių bazių reiškia rasti vidinį skirtumą ir visas situacijas, kurių atžvilgiu žinių bazės yra skirtingos, ir parodyti tuos skirtumus. Tačiau tų situacijų gali būti labai daug, o kadangi id-termuose gali būti funkcijų simboliai, tai galimų situacijų yra begalinė aibė. Tiesiogiai visas jas patikrinti neįmanoma. Parodysime, kad vietoj visų įmanomų situacijų tikrinimo pakanka patikrinti testines situacijas,

sukonstruotas iš konkretizuotų taisyklių antecedentų. Šiame straipsnyje apsiribosime Horno taisyklių atveju.

3 apibrėžtis. Formulės konkretizacija yra *elementari*, jei keitinyje, kuriuo gauta ši konkretizacija, visi kintamieji keičiami į konstantas be funkcijų simbolių.

TEOREMA. Žinių bazės KB_1 ir KB_2 yra skirtingos kurios nors situacijos S atžvilgiu tuomet ir tik tuomet, kai egzistuoja situacija Z, kuri yra vienos iš KB_1 arba KB_2 taisyklių antecedento elementari konkretizacija ir kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos.

Įrodymas.

($\Leftarrow$) Tegu egzistuoja situacija Z, kuri yra vienos iš KB_1 arba KB_2 taisyklių antecedento elementari konkretizacija ir kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos. Tuomet S ir bus Z.

($\Rightarrow$) Tegu egzistuoja situacija S, kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos ir kuri nėra vienos iš taisyklių antecedento elementari konkretizacija. Reikia įrodyti, kad egzistuoja situacija Z, kuri yra vienos iš KB_1 arba KB_2 taisyklių antecedento elementari konkretizacija ir kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos.

Tegu $d \in Diff(S, KB_1, KB_2)KB_1$. Sudarykime tokią konkretizuotų KB_1 taisyklių seką $T=[t_0, \ldots, t_n]$, kad: (a) kiekvienam i $(0 \le i \le n)$ ir kiekvienam taisyklės t_i antecedento elementui A_{ij} egzistuoja $k<j$ toks, kad A_{ij} yra t_k konsekventas, arba $A_{ij} \in S$ (t.y. taisyklės susietos tarpusavyje į grandinę), (b) d yra taisyklės konkretizacijos t_n konsekventas. Atskirai išnagrinėkime atvejus, kai keitiniuose nebuvo naudojami funkcijų simboliai ir kai jie buvo naudojami.

1) Išnagrinėkime atvejį, kai funkcijų simboliai keitiniuose nebuvo naudojami. Tuomet visos konkretizacijos taisyklių sekoje T yra elementarios. Tarkime, kad neegzistuoja situacija Z, kuri yra vienos iš KB_1 arba KB_2 taisyklių antecedento elementari konkretizacija ir kurios atžvilgiu žinių bazės KB_1 ir KB_2 yra skirtingos. Tuomet kiekvienos taisyklės t_i antecedento išvados bus tos pačios abiejose žinių bazėse. Tai reiškia, kad d generuojamas ir žinių bazėje KB_2. O tai prieštarauja mūsų prielaidai, kad $d \in Diff(S, KB_1, KB_2)KB_1$. Taigi, teorema šiuo atveju įrodyta.

2) Išnagrinėkime atvejį, kai keitiniuose buvo naudojami funkcijų simboliai. Analogiškai pirmam atvejui įrodome, kad egzistuoja antecedento konkretizacija $A\sigma$ (šiuo atveju nebūtinai elementari), kurios atžvilgiu mūsų žinių bazės yra skirtingos. Tegu $A\sigma$ nėra elementari konkretizacija, t.y. keitinyje σ naudojamas pakeitimas $X / f(\vec{a})$. Parodykime, kad ją galima pakeisti elementaria konkretizacija. Antecedentui A pritaikykime keitinį θ, kuris skiriasi nuo keitinio σ tik tuo, kad vietoj $X / f(\vec{a})$ jame naudojama X/a. Iš naujo sugeneruokime modelį. Išvadų žinių bazėse gali sumažėti, kadangi iš modelio generavimo sekos bus eliminuotos tos taisyklės, kurių nekonkretizuoto varianto antecedente buvo funkcija f ir todėl nebuvo pritaikytas keitinys $X / f(\vec{a})$.

Pavyzdžiui, jei $KB_1=\{q(X)\leftarrow p(X);\ r(X)\leftarrow q(f(X))\}$, tai $KB_1\cup\{p(f(a))\}\vdash\{q(f(a)),\ r(a)\}$, o $KB_1\cup\{p(a)\}\vdash\{q(a)\}$, kadangi antra taisyklė nesuveiks. Išnagrinėkime du atvejus:

2.1) Jokios taisyklės nebuvo eliminuotos iš modelio generavimo. Tuomet keitinys σ buvo "perduodamas" taisyklių sekoje, ir todėl jį galima vėl įstatyti vietoj keitinio θ. Tarkime, žinių bazės KB_1 ir KB_2 nėra skirtingos situacijos $A\theta$ atžvilgiu. Tuomet atstatę keitinį σ gausime $DiffInt(A\sigma, KB_1, KB_2)=\varnothing$. Tačiau tai prieštarauja teiginiui, kad žinių bazės yra skirtingos situacijos $A\sigma$ atžvilgiu.

2.2) Iš modelio generavimo buvo eliminuotos tam tikros taisyklės. Analogiškai kaip 2.1 atveju gauname, kad be šių taisyklių $DiffInt(A\sigma, KB_1, KB_2)=\varnothing$. Taigi, jos yra skirtumo šaltiniai ir galima modelio generavimą pradėti nuo jų antecedentų. Kaip jau buvo pasakyta, šiose taisyklėse keitinyje sumažėja funkcijų taikymo skaičius. Tuo atveju, jei keitinyje neliko funkcijų simbolių, mes turime pirmą atvejį, kuriuo teorema jau įrodyta. Jei dar liko funkcijų ženklų, tai vėl taikoma antro atvejo pradžioje aprašyta procedūra, kol funkcijų simbolių nebelieka.

Pateiksime skirtumo tarp teisinių žinių bazių nustatymo algoritmą.

Skirtumo tarp teisinių žinių bazių nustatymo algoritmas

Įvestis. Žinių bazės KB_1 ir KB_2.

Išvestis. Testinių situacijų Z, kurioms $Diff(Z, KB_1, KB_2)\neq\varnothing$, aibė. Skirtumo aibės $DiffInt(KB_1, KB_2)$, $Diff(Z, KB_1, KB_2)$. Taisyklių aibės, kurių pagalba gautas kiekvienas faktas iš skirtumo aibės.

1 žingsnis. Atliekamas vidinis modelio generavimas ir gaunama vidinio skirtumo aibė $DiffInt(KB_1, KB_2)$. Jei $DiffInt(KB_1, KB_2)\neq\varnothing$, kiekvienam $d\in DiffInt(KB_1, KB_2)$ gaunama tokia konkretizuotų taisyklių seką $T=[t_0, \ldots, t_n]$, kad: (a) kiekvienam i $(0\le i\le n)$ ir kiekvienam taisyklės t_i antecedento elementui A_{ij} egzistuoja $k<j$ toks, kad A_{ij} yra t_k konsekventas (t.y. taisyklės susietos tarpusavyje į grandinę), (b) d yra taisyklės konkretizacijos t_n konsekventas, (c) skirtingų taisyklių konsekventai nesutampa (kad nebūtų ciklų).

2 žingsnis. Sudaroma testinių situacijų aibė. Ji sudaroma remiantis aukščiau pateikta teorema. Konkretizuojant kiekvienos taisyklių iš KB_1 ir KB_2 antecedentus, kintamojo vardas, kuris kaip ir Prologe prasideda didžiąja raide, keičiamas į konstantos vardą, pavyzdžiui, tiesiog keičiant didžiąsias raides į mažąsias. Jei gauta konstanta priklauso Herbrand'o universumui, t.y. jau naudojama žinių bazėje, tai prie jos vardo pridedamas ženklas, kad gautume unikalų vardą (pavyzdžiui, vietoj "abc" gauname "abc1").

3 žingsnis. Kiekvienai testinei situacijai Z ir kiekvienam faktui iš $Diff(Z, KB_1, KB_2)$ gauname taisyklių sekas, kaip 1-ame žingsnyje.

Pastebėkime, kad algoritmas yra baigtinis net ir tuomet, kai naudojami id-termai su funkciniais simboliais, kadangi konkretizavimui mes naudojame keitinius be funkcijų simbolių. Žinoma, jei programoje yra rekursija, gali gautis begalinis modelio generavimas.

Teisinių žinių bazių sujungimas

Panagrinėkime antrą tesinių žinių bazių integravimo aspektą, kai tikslas yra sujungti skirtingų autorių paruoštus teisinio dokumento fragmentus, užtikrinant gauto dokumento vidinį suderinamumą. Panašus uždavinys iškyla ir tuo atveju, kai reikia integruoti naują įstatymą į veikiančią įstatymų sistemą, užtikrinant visos sistemos suderinamumą.

Suderinamumo sąlygos, arba *ribojimai*, pas mus yra taisyklės su tuščiu konsekventu, t.y.

$\leftarrow B_1, \ldots, B_n$,

kur B_1 & ...& B_n, yra neleistina situacija. Suderinamumo sąlygų pavyzdžiai:

$\leftarrow O[A->V1], O[A->V2], V1 \neq V2$

$\leftarrow$*baudžiamas*(X), *nepakaltinamas*(X), X : *asmuo*

Pirma sąlyga susijusi su F-logikos sintakse ir semantika ir ja nusakomas apribojimas, kad vienareikšmis atributas negali turėti dvi skirtingas reikšmes. Antra sąlyga nusakomas dalykinės srities apribojimas, kad nepakaltinamas asmuo negali būti baudžiamas.

Skirtingai nuo aukščiau nagrinėto teisinių dokumentų palyginimo, sujungimo atveju nepakanka gauti išvadas iš konkretizuotų taisyklių antecedentų ir patikrinti išvadų suderinamumą. Pademonstruosime tai pavyzdžiu.

2 pavyzdys.

$KB_1=\{p(X) \leftarrow q(X)\}$

$KB_2=\{p(X) \leftarrow r(X)\}$

ribojimas: $\{\leftarrow p(X), p(Y), X \neq Y\}$

Kiekvienoje žinių bazėje atlikę modelio generavimą pagal testines situacijas, suformuotas iš konkretizuotų taisyklių antecedentų, gausime:

$KB_1, q(x) \vdash \{q(x), p(x)\}$ $KB_1, r(x) \vdash \{r(x)\}$

$KB_2, q(x) \vdash \{q(x)\}$ $KB_2, r(x) \vdash \{r(x), p(x)\}$

Iš tos pačios situacijos ir skirtingų žinių bazių gautos išvados neprieštarauja suderinamumo sąlygai. Tačiau išvados iš situacijos $\{q(a), r(b)\}$ pirmoje žinių bazėje yra $\{p(a)\}$, o antroje – $\{p(b)\}$, ir pagal suderinamumo sąlygą jos yra nesuderintos.

Taigi, testines situacijas reikia parinkti kitaip.

Žinių bazių integravimas daugiausia nagrinėtas kaip jų sąveika, kai tikslas yra gauti išvadą iš kelių autonominių žinių bazių, naudojant tam tikrą konflikto sprendimo mechanizmą [3]. Taip pat buvo nagrinėtas koncepcinių schemų integravimas [4], tačiau jose nenaudojamos taisyklinės žinios. Žinių bazių integravimas į vieną bazę buvo nagrinėtas C. Baral ir jos kolegų darbuose [5, 6, 7]. Jų siūlomame algoritme naudojama SLDNF įrodymo procedūra, kurios pagalba gaunamos ribojimų (integrity constraints) konkretizacijos, kurias pažeidžia žinių bazių junginys. Nesuderinamumas išsprendžiamas, konstruojant disjunktyvinę žinių bazę. Pavyzdžiui, jungiant $KB_1=\{p(X)\leftarrow q(X);\ r(a)\}$ ir $KB_2=\{p(X)\leftarrow r(X);\ q(b)\}$ su suderinamumo sąlyga $\{\leftarrow p(X),\ p(Y),\ X\neq Y\}$ gaunama $KB=\{p(a)\vee p(b);\ r(a);\ q(b);\ p(X) \leftarrow q(X),\ X\neq b;\ p(X) \leftarrow r(X),\ X\neq a\}$. Teisinėms žinių bazėms toks sujungimas ne visai tinka dėl to, kad nesuderinamumą reikia spręsti ne per disjunktyvumą, o pagal eksperto pasirinkimą, ir dėl to, kad mums reikia atsižvelgti ne tik į žinių bazės kaip uždaros teorijos suderinamumą, bet ir į suderinamumą, papildžius teoriją

naujais faktais. Pavyzdžiui, aukščiau gautą teoriją *KB* papildžius faktais $\{q(c), r(d)\}$ bus pažeistos suderinamumo sąlygos. Tokio pažeidimo galimybė turi būti pateikta teisininko-eksperto įvertinimui.

Mes siūlome tokį algoritmą:

Teisinių žinių bazių sujungimo algoritmas

Įvestis. Žinių bazės KB_1 ir KB_2, suderinamumo sąlyga $\leftarrow B_1, \ldots, B_n$.

Išvestis. Žinių bazė KB, į kurią įtrauktos pradinių žinių bazių taisyklės, kurios nėra nesuderinamumo šaltinis. Situacijos ir taisyklės, kurios gali būti nesuderinamumo šaltiniais.

1 žingsnis. Sudaromos visos įmanomos taisyklių iš $KB_1 \cup KB_2$ kombinacijos $T=\{C_1 \leftarrow A_1, \ldots, C_n \leftarrow A_n\}$, kur $C_i\sigma_{1i} = B_i\sigma_{2i}$. Jei tokių kombinacijų nėra, tai suderinamumo sąlyga niekada negali būti pažeista, ir teisinių žinių bazių sujungimo rezultatas bus žinių bazė, sudaryta iš visų pradinių bazių taisyklių.

2 žingsnis. Konkretizuojamas kiekvienas $A_i\sigma_{1i}$ $(0 \le i \le n)$ taip pat, kaip žinių bazių palyginimo algoritmo 2-ame žingsnyje ir gaunami keitiniai θ_i $(0 \le i \le n)$ tokie, kad $A_i\sigma_{1i}\theta_i = A_{1i}, \ldots, A_{m_i i}$, kur $A_{1i}, \ldots, A_{m_i i}$ yra faktai. Tokių faktų aibė $(0 \le i \le n)$ yra situacija, pažeidžianti suderinamumo sąlygą, ir pateikiama teisininko-eksperto įvertinimui. Į kiekvieną $\leftarrow A_{1i}, \ldots, A_{m_i i}$ galima žiūrėti kaip į suderinamumo sąlygą ir kartoti 1-ąjį ir 2-ąjį žingsnius kol yra tinkamų taisyklių.

3-asis žingsnis. Į jungtinę žinių bazę įtraukiamos taisyklės, kurios nebuvo naudojamos sudarant taisyklių kombinaciją *T*.

Išvados

Teisinių dokumentų integravimo uždavinį galima iš dalies automatizuoti, formalizavus juos ir taikant straipsnyje siūlomus žinių bazių integravimo metodus. Straipsnyje įrodyta, kad žinių bazių palyginimui pakanka tikrinti elementarias (t.y. be funkcijų simbolių) taisyklių antecedentų konkretizacijas ir tai leidžia išvengti begalinio funkcijų taikymo. Skirtingai nuo teisinių žinių bazių lyginimo, jų sujungimui reikalingas ne atskirų antecedentų tikrinimas, o neleistinų situacijų, kurias nusako suderinamumo sąlygos, generavimas. Straipsnyje pateikti pagrindiniai teisinių žinių bazių palyginimo ir sujungimo algoritmai.

LITERATŪRA

[1] M. Kifer, G. Lausen, 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

[2] A. Čaplinskas, L. Paliulionienė, S. Sinkevičiūtė. Project NEMESIS: Knowledge-based aspects in law engineering systems. DEXA'95 Workshop Proceedings, Norman Revell, A Min Tjoa (eds.), 1995, p. 413-421

[3] V.S. Subrahmanian. Amalgamating Knowledge Bases. ACM Transactions on Database Systems, Vol.19, 1994, No.2, p. 291-331

[4] A.P. Ambrosio, E. Métais, J.-N. Meunier. The linguistic level: Contribution for conceptual design, view integration, reuse and documentation. Data & Knowledge Engineering, Vol. 21, 1997, p. 111-129

[5] C. Baral, J. Minker, S. Kraus. Combining Multiple Knowledge Bases. IEEE Transactions on Knowledge and Data Engineering, June 1991, Vol. 3, No. 2, p. 208-221

[6] C. Baral, S. Kraus, J. Minker, V. S. Subrahmanian. Combining Knowledge Bases Consisting of First Order Theories. Computational Intelligence, 1992, Vol. 8, No. 1, p. 45-71

[7] C. Baral, S. Kraus, J. Minker and V. S. Subrahmanian. Combining Default Logic Databases. International Journal of Intelligent and Cooperative Information Systems, 1994, Vol. 3, No. 3, p. 319-348

Logical derivation and integrating legal knowledge bases: basic algorithms

L. Paliulionienė

The article deals with two aspects of integrating legal knowledge bases. One aspect is to compare and detect differences of variants of the legal document being prepared or of documents of different countries in the same legal field. Another aspect is to combine fragments of a legal document prepared by different legislators. We formalise and store legal documents as knowledge bases, so our task turns to integrating knowledge bases. In this article we present comparison and combining algorithms based on derivation by model generation.