Liet. matem. rink, **45***, spec. nr.*, 2005, 160–166

Duomenų algoritminių priklausomybių ir jų savybių kaita reliacinėse aibėse

Birutė PLIUSKUVIENĖ, Petras ADOMĖNAS (VGTU)
el. paštas: birute.pakalniskyte@fm.vtu.lt

Reziumė. Šiame straipsnyje pateikiamas reliacinio modelio praplėtimas nauju būdu, t.y. algoritminių priklausomybių tarp atributų, kurie yra pradinių duomenų aibėse, suformuotose konkretaus taikomojo uždavinio sprendimui, realizacija. Apibrėžiama algoritminių priklausomybių samprata, klasifikavimas, bendrosios išraiškos ir kaita.

Raktiniai žodžiai: algoritminės priklausomybės, reliacinė aibė, atributas, identifikavimo modelis, kortežas, domenas.

Įvadas

Sprendžiant taikomojo pobūdžio uždavinius kyla problemų, nes kintant probleminiai sričiai dažnai tenka keisti pradinių duomenų struktūras ir turinį bei sprendžiamų uždavinių algoritmą. Kadangi iš anksto labai sunku arba neįmanoma numatyti būsimų pakeitimų, tai tenka taikomuosius uždavinius projektuoti, programuoti bei įjungti į veikiančias sistemas iš naujo. Ir tai neretai keičia taikomųjų uždavinių algoritmus. Todėl yra kuriamos adaptyvios duomenų apdorojimo sistemos, kurių tikslas – mažinti uždavinio sprendimo sutrikimus, kai keičiasi duomenys ir jų apdorojimo algoritmai. Kad būtų galima sukurti adaptyvią duomenų apdorojimo sistemą yra būtina turėti didelės apimties ir įvairaus pobūdžio duomenų struktūras, kuriose tiek pradiniai duomenys, tiek duomenys po apdorojimo yra pateikiami tokiose pat struktūrose, kurios išreiškiamos reliacinėmis aibėmis (RA). Šiomis duomenų struktūromis galime pateikti pradinius duomenis bei rezultatų duomenis apie realaus pasaulio objektus, reiškinius, procesus ir t.t. [1]

Kadangi daugelio taikomojo pobūdžio uždavinių sprendimui reikia, kad duomenys apdorojimui jiems būtų pateikti tam tikra tvarka arba eile, tam tikslui mes pateikiame reliacinės aibės identifikavimo modelį, kuriame yra visi reikalingi identifikatoriai. Šis duomenų struktūrų identifikavimo modelis, iš duomenų visumos, išreiktos reliacinėmis aibėmis, suteikia galimybę nesudėtingai surinkti duomenis konkretaus uždavinio sprendimui reikiama eile ir reikiamą jų kiekį. Tai leidžia kontroliuoti duomenų išsamumą, sutrumpinti uždavinio sprendimo laiką ir sąnaudas, nes bet kokiam taikomajam uždaviniui, kurio pradiniai duomenys ir duomenys po apdorojimo yra reliacinės aibės, tinka tas pats modelis [2].

Suformavus pradinių duomenų aibę ar aibes reikalingas konkretaus taikomojo uždavinio sprendimui, tą uždavinį galime spręsti kaip tam tikrą algoritminių priklausomybių tarp duomenų realizaciją, kadangi tarp atributų atsiranda algoritminės prik-

lausomybės. Vadinasi, šį uždavinį sprendžiant šie atributai yra siejami, t.y. vienas nuo kito priklauso. Vienam keičiantis gali keistis ar nesikeisti ir kiti atributai. Be to, pastarosios algoritminės priklausomybės gali kisti. Vienam projektuojamam apdorojimui jos gali būti vienokios tarp tų pačių atributų, kitam – jas gali reikėti atrinkti panašiai.

Algoritminių priklausomybių klasifikavimas

Teoriniu požiūriu algoritmine duomenų priklausomybe laikomas algoritmas, kuris nurodo, kaip apdorojami duomenys sprendžiant reikiamą uždavinį, kuris yra toje pačioje RA. Jeigu duomenys yra skirtingose RA, tai būtina atlikti išskirtinio pobūdžio duomenų transformacijos algoritminę priklausomybę. Pastarosios priklausomybės algoritmo realizacija iš bet kokio RA rinkinio transformuoja į vieną reikiamą konkretaus uždavinio pradinių duomenų RA. Duomenys šioje duomenų apdorojimo sistemoje faktiškai yra RA atributų reikšmės – c_{ij}. Kiekvienai atributo reikšmei arba bet kuriam RA reikšmių poaibiui parenkama tvarkioji algoritminių priklausomybių aibė, kurios elementų algoritmų realizavimo programiniai moduliai ir atlieka duomenų apdorojimą:

$$< f_{ij} > \rightarrow < c_{ij} > \rightarrow < c'_{ij} >,$$

čia $< c'_{ij} >$ yra apdorojimo rezultatas.

Jeigu sprendžiamam duomenų apdorojimo uždaviniui prireikia naujų algoritminių priklausomybių ir jas realizuojančių modelių, nes turima jų visuma negali atlikti tam tikrų uždavinio sprendimui reikalingų veiksmų, tada sukuriama nauja algoritminė priklausomybė ir ją realizuojantis modelis, kurie įtraukiami į jau turimas algoritminių priklausomybių ir programinių modelių aibes. Nauja algoritminė priklausomybė kartu su kitomis apima ir naujojo uždavinio sprendimo algoritmo realizavimą. Tad galime teigti, kad visų algoritminių priklausomybių klasių aibė bei algoritminių priklausomybių skaičius klasėje yra atviras.

Algoritminės priklausomybės (AP) gali būti suskirstytos į šešias klases:

1) RA transformacijų AP – f^t. Ši klasė nustato operaciją, kuri įgalina iš jau suformuotų konkretaus uždavinio sprendimui pradinių duomenų aibių perkelti norimas atributų reikšmes į vieną bendrą aibę. Esant reikalui transformacijos metu galime keisti duomenų aibių struktūrą ir turinį.
2) Skaičiavimo-infologinės AP – f^{si}. Ši klasė yra skirstoma į dvi glaudžiai tarp savęs susietas, bet atskiras grupes. Pirma grupė naudojama įvairiems aritmetiniams veiksmams aprašyti, gaunant ir tų veiksmų rezultatus. Antroji AP grupė naudojama įvairių aritmetinių veiksmų rezultatams sugretinti, tikrinant įvairių procesų suderinamumą bei korektiškumą.
3) Skaičiavimo AP domenuose – f^s. Skaičiavimo-infologinės AP taikomos atributų reikšmėms RA kortežuose, todėl yra apibrėžiamos skaičiavimo AP, kurios taikomos tik RA domenams. Jeigu reikia atlikti domenuose aritmetinius veiksmus, tai įvedamas papildomas specialių kortežų raktų bei įvairių atributų reikšmių sumų domenas. Tokiu atveju atributų reikšmių išdėstymas ir kortežų skaičius gali būti įvairus.
4) Nuosavos atributų reikšmių AP – f^a. Tai grupė labai įvairių, gerokai viena nuo kitos besiskiriančių algoritminių priklausomybių, kurios gana sąlyginai nuolatinės.

Tai reiškia, kad konkrečiai atributo reikšmei dalyvaujant skirtingų uždavinių sprendimuose atributo reikšmė gali būti kitokia. Net ir tokią atributo reikšmės algoritminę priklausomybę, kaip atributo reikšmė – skaičius ar tekstas, negalime traktuoti kaip nuolatinę. Nes jeigu atributo reikšmę apibrėžtume kaip tekstą, tada ši reikšmė negalėtų dalyvauti aritmetinėje operacijoje. Vieną uždavinį sprendžiant gali reikėti, kad turimos atributų reikšmės būtų traktuojamos kaip skaičiai, o kitą – kaip tekstas.

5) Programinių ėjimų AP – f^p. Daugelis algoritminių priklausomybių naudoja duomenis, esančius įvairiuose RA adresuose, todėl būtina programinį ėjimą pradėti reikiamame adrese ir tęsti reikiama kryptimi. Kitu atveju, duomenų apdorojimo procesas tampa nekorektiškas arba neįmanomas. Visi programiniai ėjimai algoritminėse RA (ARA) skirstomi į nuoseklius ir nenuoseklius. ARA yra aibė, sudaryta iš atributų reikšmių algoritminių priklausomybių identifikatorių ir vienareikšmiškai lemia atributų reikšmių dalyvavimą tam tikroje duomenų apdorojimo operacijoje.
6) Išorinės AP – f^i. Išorinės algoritminių RA priklausomybės reguliuoja apdorojamų duomenų ARA tarpusavio ryšius su gretimomis ARA. Šie ryšiai neatsiejami nuo apdorojamųjų duomenų RA.

Keletas pagrindinių algoritminių priklausomybių klasių detaliau aprašytos kituose skyriuose.

RA transformacijų algoritminės priklausomybės

RA transformacijos – tai viena iš algoritminių priklausomybių rūšių ir žymimos f^t. Aibių transformacijos išreiškiamos formule, kuri gali pakeisti reliacinės algebros operacijas, o daugelyje atvejų gali ir viršyti jų galimybes. Kadangi atliekant transformaciją gali būti realizuojami aritmetiniai bei loginiai veiksmai, todėl duomenis priimanti RA gali būti sudaryta visiškai arba dalinai iš visai naujų atributų reikšmių c_{ij}, kurių duomenų šaltinyje nebuvo. Esant reikalui galime keisti ir duomenų aibių struktūrą ir turinį [3].

Kaip jau buvo minėta, RA transformacija yra operacija, kuri įgalina perkelti atributų reikšmes iš vienos aibės į kitą. RA, iš kurios transformuojami duomenys, vadinama duomenų šaltiniu, o į kurią transformuojami – duomenis priimančia RA. Jeigu priimančios RA duomenų koordinates joje pažymėsime i ir j, o duomenų šaltinio koordinates k ir l, tai duomenys bus transformuojami iš adresų $< k/l >$ į adresus $< i/j >$. RA adresai, iš kurių imami duomenys, ir RA adresai, į kuriuos įvedami duomenys, turi būti vienodai nutolę nuo savosios RA pradžios. RA adresams pažymėti pakanka l ir j, t.y. parodyti tik c nutolimą nuo kortežo pradžios.

RA transformacijos pagal semantikos algoritmus gali būti suskirstytos į besąlygines, sąlygines, sąlygines ištisines, sąlygines ciklines, rezultatines.

Visose išvardintose transformacijose, išskyrus besąlyginę, sąlygos duomenims palyginti nurodomos simboliais: $=, \neq, >, <, \geqslant, \leqslant, \wedge, \vee, \neg$.

Sąlyginėse transformacijose yra lyginamas adresų turinys ir transformacija atliekama tik tada jeigu sąlyga patenkinta.

Transformacijų adresai RA gali būti kelių rūšių:

– adresai, iš kurių imami duomenys $< k/l >$;
– adresai, į kuriuos įnešami duomenys $< i/j >$;
– adresai, kurių turinys sulyginamas pagal nustatytą sąlygą $\{k/l\}$, $\{i/j\}$.

Šie adresai jungiami į paprastąsias aibes, o ne į tvarkiąsias, nes palyginamųjų adresų skaičiai gali labai skirtis. Pavyzdžiui, vieno adreso turinį galime lyginti su dideliu skaičiumi šaltinio adresų ir transformaciją atlikti jeigu bus patenkinta sąlyga. Šiuo atveju šaltinio adresų išdėstymo eilė visiškai neturi reikšmės.

Taigi transformacijos formulės adresinė struktūra bus

$$\{k/l\}^n < k/l >^n \xrightarrow{\Sigma} < i/j >^n \{i/j\}^n.$$

Čia lyginami adresų $\{i/j\}$ ir $\{k/l\}$ turiniai; Σ apibrėžtas vienas iš anksčiau pateiktų lyginimo sąlygos simbolių. Kai sąlyga patenkinta duomenys pernešami iš adresų $< k/l >$ į adresus $< i/j >$. Rodykle nurodyta duomenų transformacijos kryptis. Indeksas n reiškia aibės eilės numerį. Vienu metu šie indeksai gali būti visi skirtingi arba visi vienodi. Jei n reikšmės skirtingos, tai duomenų lyginimas, kad būtų įvykdyta sąlyga Σ, atliekamas vienose aibėse, o duomenų transformacija – kitose. Jei n reikšmės vienodos, tai duomenys lyginami ir transformuojami toje pačioje RA, t.y. atliekamas vienos RA struktūros (duomenų išdėstymo ir/arba kiekio) pakeitimas.

Duomenų šaltinis gali talpinti neribotą skaičių RA ir įvairių schemų. Kadangi tokio duomenų šaltinio panaudojimas atliekamas viena transformacijos formulės išraiška, todėl formulėje reikia turėti galimybę reguliuoti duomenų išrinkimą iš šaltinio ir jų įrašymą į priimančią aibę. Ženklu Δ žymimas duomenų išrinkimo reguliatorius, o duomenų įrašymo – ženklu∇. Taigi RA transformacijos formulė bus tokia:

$$f^t\Big(\Delta\{k/l\}^n < k/l >^n \xrightarrow{\Sigma} \nabla < i/j >^n \{i/j\}^n\Big).$$

Besąlyginės transformacijos atveju sąlygos Σ, formulėje nebūna, kaip ir adresų riestiniuose skliaustuose. Tad besąlyginės transformacijos formulė bus:

$$f^t\Big(\Delta < k/l >^n \nabla < i/j > \Big).$$

Šioje išraiškoje $< i/j >$ indeksas n neturi prasmės, nes duomenis priimanti RA visada yra viena.

Visose transformacijose duomenys gali būti pernešami iš vienos aibės į kitą trimis būdais:

- atliekant Dekarto transformaciją;
- atliekant transformaciją kortežuose;
- atliekant transformaciją domenuose.

Nustatome transformacijos užrašymo schemą:

$$[\text{Duomenų šaltinis} - \text{RA}] \longrightarrow [\text{Duomenis priimanti RA}]$$
$$= [\text{Transformacijos rezultatas} - \text{RA}].$$

Pateikiame Dekarto besąlyginės transformacijos RA pavyzdį, kuris atliktas pagal formulę: $f^t(<1,2>\rightarrow<3,4>)$.

$$\left[\begin{array}{cc} K & L \\ \hline a & b \\ c & d \end{array}\right] \longrightarrow \left[\begin{array}{cc} M & N \\ \hline e & f \\ g & h \end{array}\right] = \left[\begin{array}{cccc} M & N & K & L \\ \hline e & f & a & b \\ e & f & c & d \\ g & h & a & b \\ g & h & c & d \end{array}\right].$$

Tradicinės Dekarto sandaugos galimybės išplečiamos formulėmis:

$$f^t\big(<1,2>\rightarrow<1,3>\big);$$
$$f^t\big(<1,2>\rightarrow<2,4>\big);$$

nes RA rezultatas pakeičia net RA schemą:

$$\left[\begin{array}{cc} K & L \\ \hline a & b \\ c & d \end{array}\right] \longrightarrow \left[\begin{array}{cccc} M & K & N & L \\ \hline e & 0 & f & 0 \\ g & 0 & h & 0 \end{array}\right] = \left[\begin{array}{cccc} M & K & N & L \\ \hline e & a & f & b \\ e & c & f & d \\ g & a & h & b \\ g & c & h & d \end{array}\right].$$

Galimybė keisti RA schemą ypač vertinga, nes išplečia duomenų manipuliavimo galimybes, nedidinant programinių modulių skaičiaus bei sudėtingumo. Toks pavyzdys gali būti galimybė adresų nuorodomis formulėje $f^t(<2>\rightarrow<3>)$ gauti rezultatą:

$$\left[\begin{array}{cc} K & L \\ \hline a & b \\ c & d \end{array}\right] \longrightarrow \left[\begin{array}{cc} M & N \\ \hline e & f \\ g & h \end{array}\right] = \left[\begin{array}{ccc} M & N & L \\ \hline e & f & b \\ e & f & d \\ g & h & b \\ g & h & d \end{array}\right].$$

Transformacija kortеže atliekama sujungiant vienodai nutolusius nuo RA pradžios kortežus į vieną kortežą:

$$\left[\begin{array}{ccc} D & E & F \\ \hline s_1 & r_1 & z_1 \\ t_1 & u_1 & v_1 \end{array}\right] \longrightarrow \left[\begin{array}{ccc} A & D & B \\ \hline r & t & s \\ u & z & v \end{array}\right] = \left[\begin{array}{cccc} A & D & B & F \\ \hline r & s_1 & s & z_1 \\ u & t_1 & v & v_1 \end{array}\right].$$

Besąlyginės transformacijos kortеže atlikta pagal formulę:

$$f^t\big(<1,3>\rightarrow<2,4>\big).$$

Jeigu išrinkimo adresai duomenų šaltinyje ir duomenų įrašymo adresai priimančioje RA sutampa, tai turi būti atlikta besąlyginė duomenų transformacija domenuose, t.y.:

$$f^t\big(<1,2,3>\rightarrow<1,2,3>\big);$$

$$\left[\begin{array}{cccc} K & L & M & N \\ \hline a_1 & b_1 & e_1 & f_1 \\ c_1 & d_1 & g_1 & h_1 \end{array}\right] \longrightarrow \left[\begin{array}{ccc} K & L & N \\ \hline a & b & e \\ \hline c & d & g \end{array}\right],$$

$$\left[\begin{array}{ccc} K & L & M \\ \hline a_2 & b_2 & e_2 \\ c_2 & d_2 & g_2 \end{array}\right] = \left[\begin{array}{ccc} K & L & M \\ \hline a & b & e \\ c & d & g \\ a_1 & b_1 & e_1 \\ c_1 & d_1 & g_1 \\ a_2 & b_2 & e_2 \\ c_2 & d_2 & g_2 \end{array}\right].$$

Pateikti duomenų besąlyginės transformacijos pavyzdžiai tik bendrąja prasme iliustruoja galimybę transformacijos metu keisti reliacinių aibių struktūrą ir turinį. Transformacijų galimybės gali būti labai lengvai keičiamos ir plėtojamos.

Skaičiavimo-infologinės algoritminės priklausomybės

Skaičiavimo-inforloginės AP – tai algoritminės priklausomybės, kurių algoritmo realizavimo rezultatas yra skaičius ir grupė savybių, visiškai tokiu pat būdu apskaičiuojančių du skaičius ir juos tarp savęs susiejančių vienu iš simbolių: $=, \neq, >, <, \geqslant, \leqslant$.

Skaičiavimo (aritmetinės) operacijos gali būti: $+, -, :, \times$ ir kt.

Pirmoji šios klasės algoritminių priklausomybių grupė vadinama skaičiavimo AP ir žymima f^{+}, o antroji – infologinėmis AP ir žymima $f^{=}$. Bendroji pirmosios grupės algoritminių priklausomybių išraiška yra

$$f^{+}\big(A\psi A\psi[c]\psi \ldots \to A\big),$$

čia A yra atributo reikšmės adresas reliacinėje aibėje; ψ – vienas iš skaičiavimo operacijų ženklų; $\to$ – nuoroda į adresą, kur turi būti patalpintas skaičiavimo rezultatas; $[c]$ – skaitinė konstanta (konstantos panaudojimo pavyzdžiu gali būti procento skaičiavimas).

Bendrojoje infologinių algoritminių priklausomybių išraiškoje duomenų palyginimo simbolis gali būti įrašytas tik vieną kartą vietoj bet kurio skaičiavimo operacijos ženklo:

$$f^{=}(A\psi A\psi A\psi \ldots A).$$

Kadangi šioje klasėje glaudžiai siejasi skaičiavimo operacijos ir informacijos logika, atspindinti realaus pasaulio objektus ir procesus, todėl aprašytos algoritminės priklausomybės vadinamos skaičiavimo-infologinėmis priklausomybėmis.

Išvados

Apibrėžiama duomenų algoritminių priklausomybių realizacija ne tik nurodo kaip, sprendžiant tam tikrą taikomojo pobūdžio uždavinį, galime apdoroti duomenis, bet ir leidžia juos apdoroti. Tai įmanoma, kadangi kiekviena algoritminė priklausomybė

turi savo koncepcijos realizavimo algoritmą ir programinį to algoritmo realizavimo modulį. Taigi, įvairiems uždaviniams spręsti galime naudoti tuos pačius algoritminių priklausomybių algoritmus. Tai leidžia plėsti sprendžiamų uždavinių sąrašą nekuriant naujų programinių priemonių, kadangi minėtus algoritmus galime realizuoti nepriklausomais arba autonomiškais programiniais moduliais. Jeigu konkrečiam uždaviniui spręsti pritrūksta esamų algoritminių priklausomybių, tada galima plėtoti šią sistemą. Tai atliekama ne kuriant programines priemones, kurios realizuotų naujo uždavinio algoritmą tiesiogiai, bet papildant turimą sistemą naujomis algoritminėmis priklausomybėmis, kurios leistų su kitomis jau egzistuojančiomis AP išspręsti naują uždavinį. Ši sistemos papildymo galimybė dažnai duoda naujų, tiesiogiai neplanuotų galimybių. Ir kuo daugiau sistema papildoma naujomis algoritminėmis priklausomybėmis, tuo rečiau prireikia tokių papildymų.

Literatūra

1. P.G. Adomėnas, Data functional feature sets and their adaptability, in: *Proc. V East-European Conference on Aadvances in Databases and Information Systems*, Vilnius (2001), pp. 131–140.
2. B. Pakalniškytė, P. Adomėnas, Duomenų struktūrų identifikavimo modelis, kn.: *Praneš. medž. IV konf. Informacinės technologijos*: *aktualijos ir perspektyvos*, Alytus (2005), pp. 157–162.
3. P.G. Adomėnas, A. Čiučelis, Data aggregation sets in adaptive data model, *Informatica*, **13**(4), 381–392 (2002).
4. A. Binemann-Zdanowicz, Current issues in databases and information systems, in: *Proc. East-European Conference on Advances in Databases and Information Systems*, Prague (2000), pp. 307–314.
5. B. Thalheim, *Dependencies in Relational Database*, Teubner (1991).

SUMMARY

B. Pliuskuvienė, P. Adomėnas. Data algorithmic dependencies and their features vicissitude in relation sets

This article presents introduction of the new mechanism to the relation model, particularly algorithmic dependencies showing up among attributes, which are realization of concrete applicable task structured in primary data sets.

The conception of algorithmic dependencies, classification, general expressions and changes are also defined.

Keywords: algorithmic dependences, relational set, attribute, identification model, tuple, domain.