Adaptyvios duomenų modeliavimo sistemos

Turinys

1. Įvadas **2**

2. Duomenų struktūros **3**

3. Formalūs apibrėžimai **4**

4. Aibių operacijos **5**
4.1. Aibių jungimo operacija . . . 5
4.2. Aibių atimties operacija . . . 6
4.3. Dekarto sandauga . . . 6
4.4. Projekcijos operacija . . . 6
4.5. Aibių papildinys . . . 6
4.6. Selekcijos (išrinkimo) operacija . . . 7
4.7. Pjūvio operacija . . . 7

5. Funkcinės atributų reikšmių savybės **7**
5.1. Funkcinės aibės . . . 8
5.2. Duomenų identifikavimas . . . 10
5.3. Nuosavos atributų reikšmių funkcinės savybės (f^X) . . . 10
5.4. Skaičiavimo-infologinės atributų reikšmių funkcinės savybės . . . 12
5.4.1. Skaičiavimo funkcinės savybės . . . 12
5.4.2. Infologinės funkcinės savybės . . . 12
5.5. Skaičiavimo funkcinės savybės r-domenose . . . 12
5.6. Programinių ėjimų funkcinės savybės . . . 13
5.7. Duomenų transformacijos funkcinės savybės . . . 15
5.7.1. Besąlyginės transformacijos . . . 16
5.7.2. Sąlyginės transformacijos . . . 18
5.8. Išorinės funkcinės savybės . . . 21
5.9. Pagalbinės funkcinės savybės . . . 22

6. Matematinis taikomojo uždavinio modelis **22**
6.1. Duomenų pateikimo modeliavimas . . . 23
6.2. Bendrasis uždavinio matematinis modelis . . . 23

7. Operacijos `ADD`, `DEL` ir `CHANGE` **24**
7.1. `ADD` operacija . . . 24
7.2. `DEL` operacija . . . 24
7.3. `CHANGE` operacija . . . 24

1. Įvadas

Duomenų modeliavimas vyksta reliacinio duomenų modelio plotmėje.

Adaptacija - tai automatinis tam tikros informacinės sistemos prisitaikymas, tada, kai sprendžiant konkretų uždavinį keičiasi duomenų struktūros (nekiekybiškai), bei tų duomenų apdorojimo algoritmai. Tai yra efektyvu, nes nereikia programuoti naujo uždavinio.

Adaptacijos metodai tam tikra savo dalimi naudojami jau dabar, jų metodų neįvardijant: šie metodai naudojami ne kaip sistema, o kaip atskiri metodai. Šiame kurse nagrinėjamos dviejų tipu reliacinės aibės viena iš kurių turi kintamą kortežų skaičių, o kita - pastovų. Nagrinėsime funkcinės duomenų savybės: aritmetines funkcijų savybės, duomenų transformacijas aibėse ir t.t. Iš funkcinių savybių sudaromos funkcinės aibės. Kiekviena funkcinė duomenų savybė realizuojama diskrečių programinių modulių.

Taigi, funkcinėse aibėse funkcinės savybės išrikiavus tam tikra tvarka ir jos vykdant, gali būti realizuoti tam tikri programiniai modeliai, reiškiantys šių funkcinių aibių savybės ir tokių būdu sprendžiamas uždavinys. Jeigu pasikeičia duomenų savybės - keičiasi funkcinių savybių aibė ir jų programų modelio saugojimo tvarka ir uždavinys gali būti sprendžiamas toliau. Jeigu atsitinka taip, kad tam tikram uždaviniui spręsti pritrūksta funkcinių savybių, to pasekoje ir programinių modulių, tai adaptyvioji sistema elgiasi nestandartiškai: kuriama nauja funkcine savybe (pagal tas pačias taisykles), kuri aprėpia uždavinius taip, kad uždavinys, kurio iki šios funkcinės savybės sukūrimo, išspręsti buvo neįmanoma. Be to, nauja funkcinę savybė kartu su senosiomis praturtina visą funkcinių savybių šeimą naujomis galimybėmis, kuriu ankstesnių laiko momentu dar nereikėjo. Tokiu būdu adaptyvioji sistema greitai "tvirtėja" funkcinių požiūriu: plečiasi jos funkcinės galimybės ir laikui bėgant naujų funkcinių savybių atsiradimas tampa labai retu reiškinių. Apskritai, tokia sistema yra gyvybingiausia, kai duomenys pateikiami reliacinėmis aibėmis ir rezultatai irgi yra reliacinės aibės. Funkcinės savybes, surašytos į funkcinę aibę pagal atitinkamus matmenys (rangą ir aibės tipą) nebūtinai atitinka duomenų struktūros rangą. Svarbiausiai kad funkcinės savybės būtų realizuotos nustatyta tvarka. Realizacijos eilė turi būti griežtai nurodyta. Antra vertus, duomenys kurios reikia apdoroti konkrečiam uždaviniui, reikia pateikti taip pat griežtai nustatyta tvarka. Paprastai vienos funkcinės aibes pakanka patikrinti, ar vieno ir to paties tipo duomenų realizacijos aibes yra korektiškos.

Sudėtingiems uždaviniams tenka naudoti eilę funkcinių savybių $F_n\psi F_n\psi \ldots$, kur F_n - funkcinės-reliacinės aibes, o ψ - išorine funkcinė savybė, kuri nurodo tarpusavio informacinius ryšius tarp einančių greta eilutėje funkcinių aibių.

Duomenys, kurie pateikiami reliacinėmis aibėmis, gali būti labai gausūs, ir ta gausa pasireiškia ne tik aibių kiekių, bet ir jų struktūrų įvairovę. Tai kaip gi mes galime nurodyti įvairių aibių didelį kiekį? Tai atliekama šių aibių tam tikra identifikacija. Didžiajai daugumai aibių pakanka trijų identifikatorių:

- aibės struktūros identifikatoriumi A
- aibes priklausomumo subjektui (objektui) identifikatoriumi B
- laiko identifikatoriumi D

Jei šių identifikatorių nepakanka, tai reiškia kad atsiranda dvi ar daugiau aibių su vienodais identifikatoriais, bet skirtingomis reikšmėmis. Tada prie šių identifikatorių nurodomi adresai (duomenų koordinatės lentelėse), kurių reikšmės padeda papildomai identifikuoti aibes. Konkrečiam uždaviniui spręsti gali prireikti labai daug duomenų - daug reliacinių aibių. Nurodyti kiekvienos iš jų identifikatorius dažnai būtų neįmanoma. Todėl naudojamas supaprastintas duomenų identifikatoriaus užrašymo būdas, o juos reikiama eilę sustato specialios adaptyvios sistemos programos. Funkcinės duomenų savybės savo struktūra, sudėtingumu, pobūdžiu, algoritmais yra labai skirtingos. Pvz.: pateikiant sistemai duomenys patikrinti, ar duomuo yra tekstas ar skaičius? Kitos funkcinės savybės (pvz.: duomenų transformacijos) yra labai sudėtingos.

2. Duomenų struktūros

Pagrindinės duomenų struktūros yra dviejų tipų: reliacinės aibės r ir v. Šiose aibėse elementai išdėstyti tam tikra tvarka vienas kito atžvilgių ir tarp duomenų egzistuoja tam tikri santykiai arba sąryšiai, todėl aibė vadinama reliacine. (*relation* - santykis, sąryšis)

Taigi, adaptyvumo teorija bus pateikta reliacinio duomenų modelio plotmėje. Prieš pradedant nagrinėti adaptyvumo procesą, metodus, būdus, apibrėžkime pagrindines duomenų struktūrų kategorijas. Būtina prisiminti, kad reliacinė aibe praktiškai reiškia duomenų lentelę. Tai labai paprasta ir įprasta visiems duomenų konstrukcija ir net sunku įsivaizduoti tokius duomenys, kurių negal ėtume aprašyti lentele. Kita vertus, kadangi matematikoje galime lentelę traktuoti kaip aibę, mes turime seniai ir plačiai išplėsta matematikos šaka - aibių teorija. Dažnai elementų išdėstymo tvarka aibėje lemia jų semantika. Aibė r vadinsime aibė reliaciniame medyje. Ją sudaro atributai ir atributų reikšmes.

$$r = \begin{bmatrix} & A_1 & \dots & A_n \\ C_{11} & C_{12} & \dots & C_{1n} \\ C_{21} & C_{22} & \dots & C_{2n} \\ \dots & \dots & \dots & \dots \\ C_{m1} & C_{m2} & \dots & C_{mn} \end{bmatrix}; A - \texttt{atributai}\ C_{ij}, \texttt{kur} 1 \le i \le m, 1 \le j \le n$$

Šios aibės rangas n yra visada pastovus.

Jeigu atributų kiekis pasikeis - gausime kitą aibę. Visi atributai vadinami aibės schema $R(A_j)$. Skaičius m vadinamas aibės galia arba eile. r-tipo aibė tos pačios schemos bet kaip užpildoma duomenims gali turėti skirtingą eilę.

Vieno atributo reikšmės kartu su pačiu atributu vadinami šio *atributo domenu*.

$D = \frac{A_j}{C_{ik}}$, čia k - fiksuotas skaičius.

Visos aibės atributų reikšmės sudaro *aibės domeną*.

[] - tai tvarkiosios aibės ženklas. Nežiūrint į tai, kad aibė R yra tvarkioji, jos eilutės (kortežus) galime keisti vietomis ir nuo to aibė nepasikeis.

Tenka ir kortežą nuo kortežo atrinkti duomenų apdorojimo metu. Tos atributo reikšmes, kurios viename domene yra skirtingos galime laikyti *kortežo (eilutės) raktu* ir jo pagalba skirti vieną kortežą nuo kitų. Jeigu skirtingi kortežai turi vieno atributo vienodų reikšmių, tai tada naudojami du atributai.

Aibės v vadiname funkcinėmis, nes jos turi sudėtingą atributą ir formatą:

$$v = \left[\begin{array}{c|ccc} A_1 & A_2 & \dots & A_n \\ \hline B_1 & C_{11} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ B_i & \dots & C_{ij} & \dots \\ \dots & \dots & \dots & \dots \\ B_m & \dots & \dots & C_{mn} \end{array}\right];$$

Kiekvieno atributo reikšmę apibrėžia du poatribučiai: $\frac{B_i}{A_j}$. Iškyla esminis klausymas: kodėl negalime apsieiti su tokia pat aibę r, turinčia vieną kortežą? $B_1A_1\ B_1A_2 \dots B_1A_n$, $B_2A_1\ B_2A_2 \dots B_2A_n$, $\dots B_mA_1\ B_mA_2 \dots B_mA_n$ - naujas dėsnis, reikalingas todėl, kad atribute atsiskleidžia duomenų savybės, kurios lemia duomenų apdorojimo algoritmus. Taigi, sujungti aibės v atributus ir padaryti vieną kortežą paprasta. Atvirkščias reiškinys, kai atributai yra daugiareikšmiai, praktiškai neįmanomas. Todėl nukenčia atributu apdorojimo aiškumas.

Panagrinėkime sudėtingų atributų konstrukciją.

Darbo užmokestis A_1^1			
PlanuotasA_2^1		Faktiškas A_2^2	
Darbininkų A_1^3	Tarnautojų A_2^3	Darbininkų A_3^3	Tarnautojų A_4^3
$A_1^1A_1^2A_1^3$	$A_1^1A_1^2A_2^3$	$A_1^1A_2^2A_3^3$	$A_1^1A_2^2A_4^3$

Į kiekvieną atributą įeina komponente A_1^1. Jeigu aibėje r kuri nors atributo reikšmė lygi 0, tačiau tuščios reikšmės būti negali. O aibėje v gali atsirasti tuščių reikšmių.

Riedmenų techninės priežiūros ir remonto bazė			
	Remontas	Visas plotas	Atlikta vienetų
Atlikta, vnt.		0	
Patalpų plotas			0

Aibė v turi tokias savybės:

1) atributai gali būti sudėtiniai (iš dviejų ar daugiau dalių). Sudėtinio atributo dalį kartais vadiname poatributu.

2) Visada fiksuota aibės galia visiems tos pačios schemos egzemplioriams.

3) Bet kuris $\frac{B_i}{A_j}$ turi tik vieną reikšmė.

4) Atributų reikšmių skaičius yra rango ir eilės (galios) sandauga, įskaitant ir atributus, kurie turi tuščias reikšmės.

5) atributas gali turėti tuščią reikšmę

6) v aibė tenkina lygybė:

$$\begin{bmatrix} & A & B \\ C & a & b \\ D & c & d \end{bmatrix} = \begin{bmatrix} & C & D \\ B & b & d \\ A & a & c \end{bmatrix} = \begin{bmatrix} & D & C \\ A & c & a \\ B & d & b \end{bmatrix} = \begin{bmatrix} & B & A \\ D & d & c \\ C & b & a \end{bmatrix}$$

3. Formalūs apibrėžimai

1) Aibė r vadiname aibe, kuri reiškia atitikimybę tarp atributų $A_1, A_2, \ldots, A_n$ ir jų reikšmių:
C_{11} C_{12} $\ldots$ C_{1n}
C_{21} C_{22} $\ldots$ C_{2n}
$\ldots$ $\ldots$ $\ldots$ $\ldots$

2) Aibės atributų poaibį vadiname $R = R(A_j) = R(A_1, A_2, \ldots, A_n)$

3) Atributų reikšmių aibė vadinsime r-domenu. $D(r) = \mathrm{dom}(r) = \{C_{ij}\}_{1\le j\le n}^{1\le i\le m}$

4) Aibės reikšmių poaibį, kur $\gamma = b$ vadiname atributo A_b domenų. Tai reiškia, kad $\mathrm{dom}(A_b) = C_{1b}, C_{2b}, \ldots, C_{mb}$ T.y. domenas - tai aibės atributas ir visos šio atributo reikšmės. Domenas egzistuoja tik aibės atžvilgių. Pridėjus prie aibės kortežą, kurio reikšmė nurodytam atributui nebus lygi nei vienai kitai šio atributo reikšmei, gausime naują domeną.

5) Aibės r domeno poaibis, turintis fiksuotą indeksą $i = a$ dar vadinamas aibės kortežu. $t_a = t(C_{ja}) = C_{j1}, C_{j2}, \ldots, C_{jn}$, kur n - kortežas.

6) Kortežo poaibį $t_k \in t_{nj}$ vadiname raktu, jeigu $t_{k1} \neq t_{k2} \neq t_{k3}, \ldots$ (nelygios atributo reikšmės vadinamos raktais)

Aibė su viršutiniais indeksais $R^i = (r^1, r^2, r^3, \ldots)$ tai tos pačios aibės egzemplioriai. Tada $R_i = (r_1, r_2, r_3, \ldots)$ - kitos schemos aibė (kita aibė)

Dažnai atsitinka taip, kad nagrinėjant tam tikrus klausimus (teorinius), nėra skirtumo, ar nagrinėjamoji aibė yra r ar v tipo. Šiuo atvejų aibė žymima taip: $u = \begin{cases} r \\ v \end{cases}$

Dažniausiai atsitinka taip, kad aibės yra apjungtos į aibių sistemą. Tada mes sakome, kad turime aibių sistemą: $N =< u_1, u_2, \ldots, u_t >$

$<>$ - tai tvarkiosios aibės ženklas

Praktikoje dažnai susiduriame su u ir r sistemomis. Dažniausiai sistemos aibių skaičius yra nedidesnis už: $2, 3, ... < 10$

4. Aibių operacijos

Mūsų kurse aibių operacijos skirstome į dvi dalis:

1) tradicinės reliacinių aibių operacijos

2) duomenų funkcinių savybių realizacijos operacijos.

Pirmosioms operacijoms priklauso:

- reliacinių aibių jungimo operacija
- reliacinių aibių atimties operacija
- reliacinių aibių Dekarto sankirtos operacija
- reliacinių aibių selekcijos operacija (išrinkimo)
- projekcijos operacija
- papildinio operacija

Šių penkių operacijų kombinacija gali duoti kitokias operacijas, kurios gali būti labai įvairios. Atskirai išrenkamos dar dvi dažnai naudojamos operacijos:

- pjūvio
- sujungimo (skiriasi nuo jungimo operacijos)

4.1. Aibių jungimo operacija

Žymėjimas: $r_1 \cup r_2$ (tai kad indeksai yra apatiniai, reiškia jog aibės yra skirtingos).

Tegul $r_1 = \begin{bmatrix} A_1 & A_2 & A_3 \\ a & b & c \\ d & e & f \\ a & e & c \end{bmatrix}$, ir $r_2 = \begin{bmatrix} A_1 & A_2 & A_3 \\ a & d & c \\ d & e & f \end{bmatrix}$

Aibių jungimo operacijos rezultatas vadinamas sąjunga:

$$r_1 \cup r_2 = \begin{bmatrix} A_1 & A_2 & A_3 \\ a & b & c \\ a & e & c \\ d & e & f \\ a & d & c \end{bmatrix}$$

Tai reiškia, kad į aibių jungimo rezultatą pridedami tik tie kortežai, kurie nesikartoja aibėse. Jeigu vienodas kortežas yra abiejose aibėse, tai jis pridedamas tik vieną kartą. (Kartais, sprendžiant uždavinius, tarpinis rezultatas yra nesvarbus.) Šio pavyzdžio atžvilgių, turi būti: $A_1 \equiv A_4$; $A_2 \equiv A_5$; $A_3 \equiv A_6$.

4.2. Aibių atimties operacija

Atributu atžvilgiu, aibių atimties operacija yra tokia pat, kaip ir jungimo. Aibių jungimo operacija yra komutatyvi, tačiau atimties operacija - nėra komutatyvi.

$$r_1 - r_2 = \begin{bmatrix} A_1 & A_2 & A_3 \\ a & b & c \\ a & e & c \end{bmatrix}$$

Čia r_1 "turinys", r_2 - "atiminys", o rezultatas vadinamas skirtumu.

4.3. Dekarto sandauga

Pažymėjimas:

$$r_1 \times r_2 = \begin{bmatrix} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 \\ a & b & c & a & d & c \\ a & b & c & d & e & f \\ d & e & f & a & d & c \\ d & e & f & d & e & f \\ a & e & c & a & d & c \\ a & e & c & d & e & f \end{bmatrix}$$

Ši operacija yra daugiau teorinio pobūdžio. Viena iš pritaikymo sričių - tai grafika. Pvz.: taškų bei kreivių rinkinys - tai reliacinės aibės, Dekarto sandauga naudojama grafinių elementų tarpusavio sąveikoms nagrinėti.

Visos iki šiol minėtos operacijos vadinamos binarinėmis, nes operuoja dviem reliacinėms aibėmis (pjūvio bei sujungimo operacijos taip pat yra binarinės). Likusios pagrindinės operacijos (projekcija, selekcija (išrinkimas)) yra unarinės (vienanarės), nes operuoja tik viena aibe.

4.4. Projekcijos operacija

Ši operacija yra žymima taip:

$$\pi_{A_3=A_1} = \begin{bmatrix} A_3 & A_1 \\ c & a \\ f & d \\ c & a \end{bmatrix}$$

Taigi, galėtume pasakyti, kad aibių projekcijos operacija operuoja domenais ir gali dalį jų pašalinti, ir be to keisti reliacinės aibės schema (rezultate gražinama aibė, kurios schema skiriasi nuo pradinės). Pavyzdžiui, nustatoma kita atributu tvarka.

4.5. Aibių papildinys

Aibių papildinys - tai operacija, įgalinanti išskleisti aibę į aibių identifikatorius, kai tos aibės atributai yra aibių identifikavimo komponentes. Aibių struktūros identifikatoriai reiškia aibės pavadinimą bei jos schemą $< R >$.

Tegul turime aibę: $\{a_1, a_2\}$, kur a_1 ir a_2 - aibių struktūrų identifikatoriai. Taip pat turime aibę $\{b_1, b_2\}$ kuri identifikuoja subjektus. Pvz.: universiteto subjektais gali būti fakultetai, studentai ir t.t., įmonės subjektais gali būti jos padaliniai arba darbuotojai. Reikalinga dar viena aibė: $\{d_1, d_2\}$, kuri identifikuoja duomenys laiko atžvilgių ir gali reikšti laiko tarpą arba momentą. Šiuo atveju aibių papildinio operacija yra tokia:

$$\mathrm{dom}(\omega) = \begin{bmatrix} a_1 & b_1 & d_1 \\ a_1 & b_1 & d_2 \\ a_1 & b_2 & d_1 \\ a_1 & b_2 & d_2 \\ a_2 & b_1 & d_1 \\ a_2 & b_1 & d_2 \\ a_2 & b_2 & d_1 \\ a_2 & b_2 & d_2 \end{bmatrix}$$

Tokiu būdu papildinio operacija teorinių požiūrių sudaro maksimalų galimą identifikatorių skaičių. Realaus objekto identifikatorių skaičius gali būti tik mažesnis.

Pažymėtina, kad $a_1b_1d_1 \equiv a_1d_1b_1 \equiv b_1a_1d_1 \equiv b_1d_1a_1 \equiv d_1a_1b_1 \equiv d_1b_1a_1$

Nežiūrint į pateiktą tapatumą, identifikatorių viduje atskirų dalių tvarkos manipuliavimas leidžia keisti kreipimosi prie duomenų tvarką. Pavyzdžiai, jeigu identifikatorius viduje dalys eina tokia tvarka: $a_id_jb_j$, tai pirmiausia bus surinkti duomenys pagal a_i faktorių.

Naudojant šį metodą su kitais skleidimo metodais, galime konkrečiam uždaviniui reikalinga reliacinių aibių aibe pateikti reikiama eile.

Kreiptis į duomenys, esančius esančius duomenų bazėje galima pagal identifikatorių arba nustačius reikiamą reliacinių aibių adresų eilę.

4.6. Selekcijos (išrinkimo) operacija

Šios operacijos metu, atrenkami kortežai, kurie turi tam tikrus požymius. Operacijos užrašymas: $\sigma_\theta(r_1)$. Čia θ - tai išrinkimo sąlyga: $\theta : <, >, \neq, \leq, \geq, \cap, \cup, \neg, \forall, \exists, \ldots$

$$\theta_{A_1=a}(r_1) = \begin{bmatrix} A_1 & A_2 & A_3 \\ a & b & c \\ a & e & c \end{bmatrix}$$

4.7. Pjūvio operacija

Ši operacija yra dvigubos atimties operacija. Pjūvio operacijos rezultatas yra sudarytas iš abiem aibėms tapačių kortežų:

$$r_1 \cap r_2 = r_1 - (r_1 - r_2) = \begin{bmatrix} d & e & f \end{bmatrix}$$

5. Funkcinės atributų reikšmių savybės

Sprendžiant tam tikrą duomenų apdorojimo uždavinį, galima sakyti, kad atributų reikšmės įgyja tam tikras funkcinės savybės. Tai reiškia, kad viena atributo reikšmė, esanti reliacinėje aibėje, visos reliacinės aibės atributų reikšmės arba jų dalis apdorojamos pagal tam tikrus algoritmus, kurie realizuojami diskrečiais programiniais moduliais. Savaime suprantama, kad sprendžiant kitą duomenų apdorojimo uždavinį, ta pati atributo reikšmė gali būti apdorojama kitais algoritmais. Taigi, reliacinės aibės atributų reikšmių apdorojimo algoritmus galima sugrupuoti į tam tikras diskrečias algoritmų

grupes ir pavadinti (tam tikroje duomenų apdorojimo situacijoje) *funkcinėmis atributų reikšmių savybėmis* (angl.: functional feature). Paprasčiausias atributų reikšmių funkcinių savybių pavyzdys gali būti toks:

- Vieno (ir to paties) asmens konstatuojami "gimimo metai" atributo reikšmė yra tekstas, nežiūrint į tai, kad išreikšta skaičiumi.
- Ta pati atributo reikšmė - amžius, jau išreiškiama skaičiumi.

Apskritai funkcinių atributų reikšmių savybių gali būti labai daug. Mes jas suklasifikuosime. Tos klasės liks "nelūždamos", t.y. prireikus galėsime įvesti naują atributų reikšmių funkcinių savybių klase, arba praplėsti esamą klase naujomis funkcinėmis atributų reikšmių savybėmis. Šiame kurse bus nagrinėjamos šios atributų reikšmių funkcinių savybių klasės:

1) Nuosavos atributų reikšmių funkcinės savybės (f^{χ})

2) Laiko faktorių funkcinės savybės (f^{D})

3) Skaičiavimo-infologinės funkcinės savybės (f^{+}-skaičiavimo, $f^{=}$ -infologinės)

4) Skaičiavimo funkcinės savybės r-domenuose (f^{Σ})

5) Programinių ėjimų funkcinės savybės ($f^{\rightarrow}$)

6) Reliacinių aibių transformacijos funkcinės savybės ($f^{\sharp}$)

7) Pagalbinės funkcinės savybės (f^{*})

Be to yra speciali funkcinių savybių klasė. Šias savybės vadinamos išorinėmis funkcinėmis savybėmis. Šias funkcinės savybės nustato funkcinių aibių eilutės greta einančių aibių tarpusavio santykį.

Pastaba: bet kokia funkcinė savybė gali savo ruoštu būti funkcinių savybių aibe. Tai reiškia, jog vienai ir tai pačiai atributo savybei, nustatyti, apdoroti arba patikrinti gali būti naudojamas reikiamas funkcinių savybių, vykdomų reikiama tvarka, kiekis.

5.1. Funkcinės aibės

Jeigu funkcinė savybe reiškia atributų reikšmių apdorojimo algoritmą, tai tų algoritmų panaudojimas turi būti tam tikru būdu organizuotas ir struktūrizuotas. Funkcinės savybės konkrečiam duomenų apdorojimo uždaviniui spręsti jungiame į reliacinę aibę, vadinama *funkcine aibe*, arba funkcinių savybių aibė.

Tegul aibei r, funkcinių savybių aibė yra F_r, o aibei v - F_v. Kai nėra skirtumo, ar funkcinių savybių aibė yra r ar v, tada rašome: $u \rightarrow F_u$. Funkcinių savybių aibių sistemai: $N = u_1, u_2, \ldots$ gausime funkcinių aibių sistemą F_N.

Iš esmės, funkcinės aibės skirtos reliacinių aibių r duomenims apdoroti, sudaro funkcinę aibę $F_r =< f, f, \ldots >$. Šiuo atveju kiekvienas $f =< f', f', \ldots >$, t.y. kiekviena funkcinė savybė savo ruoštu yra sudaryta iš tam tikrų funkcinių savybių. Taip atsitinka todėl, kad reliacinė aibė r yra tokia:

$$\begin{bmatrix} A & A & \ldots & A \\ c & c & \ldots & c \\ c & c & \ldots & c \\ \ldots & \ldots & \ldots & \ldots \\ c & c & \ldots & c \end{bmatrix}$$

, Tai yra vienas atributas gali turėti daug atributų reikšmių, bet aibėje r tas pats atributas turi skirtingas reikšmes, bet tas reikšmes turi tas pačias funkcinės savybės. Tai yra atributo funkcinė savybė tam tikra prasme yra vektorius.

Visiškai kitokia funkcinių savybių aibė yra skirta reliacinei aibei v:

$$v = \begin{bmatrix} & A & A & \dots & A \\ B & c & c & \dots & c \\ B & c & c & \dots & c \\ \dots & \dots & \dots & \dots & \dots \\ B & c & c & \dots & c \end{bmatrix}$$

Kaip matyti iš reliacinės aibės v struktūros, kiekviena atributo reikšmė yra originali ir turi tik vieną reikšmę ($A_i B_j = c$). Todėl tos reikšmės apdorojimas turi būti charakteringas tik jai. Todėl funkcinių savybių aibė F_v yra dvimate, o ne vektorius:

$$F_v = \begin{bmatrix} f & f & \dots & f \\ f & f & \dots & f \\ \dots & \dots & \dots & \dots \\ f & f & \dots & f \end{bmatrix}$$

, čia taip pat kiekvienas $f =< f', f', \dots >$.

Dėl to, kad būtų paprasčiau, čia mes pateikiame funkcinių savybių ir duomenų reliacinės aibės, turinčias vienodą rangą, o aibių b atžvilgių - ir vienodą galią. Bendriausiu atveju nėra būtina, kad aibių rangas ir/arba galia sutaptu.

Vienos funkcinės aibės funkcinių savybių, kaip taisyklė, pakanka tik duomenų įkėlimui į kompiuterio atmintį, korektiškumo patikrinimui, automatiniam klaidos ištaisymui arba pranešimui galutiniam vartotojui, kad jis ištaisytų klaidą. Kaip gi tada išspręsti realų sudėtingą uždavinį?

Dažniausiai tenka apdoroti įvairių schemų ištisą reliacinių aibių eilę: $u, u, \dots$. Šią eilę galima pavadinti duomenų šaltinių. Tada aibėms uždedami tam tikri reikalavimai, kuriuos sudaro funkcinių aibių eilutę: $F_n \varphi F_n \varphi \dots$. Šioje eilutėje greta einančios dvi funkcinės aibės sieja speciali išorinė funkcinė savybė: φ.

Eilė $u, u, \dots$ yra apdorojama eilės $F_n \varphi F_n \varphi \dots$, rezultate gaunant u, arba r, arba v, arba N.

Visiškai nieko negalima būtų pasiekti, jeigu konkrečios $u = \begin{cases} r \\ v \end{cases}$ neturėtu originalaus identifikatoriaus, kuris įgalintu atskirti vieną reliacinę aibė nuo kitos.

Reliacinei aibei identifikuoti paprastai pakanka trijų identifikatorių:

- struktūros kodo a
- subjekto kodo b
- laiko faktoriaus kodo d

Struktūros kodas faktiškai reiškia reliacinės aibės schemos kodą.

Pvz. 1. *pasas (neužpildytas) atitinka schemą (a)*
Kam priklauso užpildytas pasas - subjektas (b)
Laikas, kada pasas yra išduotas - d

Laiko faktorius gali būti dviejų rūšių: *laiko momentas* ir *laiko tarpas*.

Pvz. 2. *Ataskaita 2001 metų sausio 1-ai dienai - laiko momentas.*
Ataskaita už 2000 metus - laiko tarpas

5.2. Duomenų identifikavimas

Duomenų identifikavimą reikia atlikti tam, kad galima būtų priskirti konkrečioms r ir v aibėms atitinkančius F_r ir F_v. Be to, jeigu negalėsime atskirti vieną F_u aibė nuo kitos, tai negalėsime spręsti uždavinio. Todėl būtina kiekvieną aibę r, v, F_r, F_v identifikuoti.

Identifikavimas atliekamas sukuriant kiekvienos iš aibių identifikatorių, kuris sudarytas iš trijų dalių aibėms r ir v, o aibėms F_r ir F_v pakanka vienos identifikatoriaus komponentės. Jeigu aibėms r ir v identifikuoti nepakanka trijų komponenčių, tai sukuriamas papildomas identifikatorius.

Aibėms identifikuoti naudojamos šios komponentės:

- a - schemos identifikatorius
- b - subjekto identifikatorius
- d - laiko momento (laiko tarpo) identifikatorius

Jeigu aibe r (arba v, t.y. aibė u) yra išreiškiami normatyviniai, žinyniniai duomenys, dažniausiai pakanka tik a identifikatoriaus. Jeigu tie duomenys naudojami lokaliai, reikia naudoti ir b kodą. Didžiąją daugumą dokumentų pakanka identifikuoti kodu sudarytų iš trijų dalių.

Tačiau kartais pasitaiko atvejų, kai visi šie identifikatoriai yra vienodi, tada prie kodo $a\ b\ d$ pridedami identifikatoriai: $a\ b\ d\ (i|j)$. Čia $i|j$ - tai eilutė/stulpelis - skirtingų duomenų adresas.

Funkcinėms aibėms pakanka vieno kodo - schemos kodo.

Bendriausių atvejų $R =< A_r >< B_j >< D_k >$. Prie duomenų išraiškų atributų ir jų reikšmių galime priskirti ir identifikatorius. $(abd \equiv adb \equiv bad \equiv bda \equiv dab \equiv dba)$

Taigi, galima sudaryti papildinio operacija, kuri gali išplėsti pagal tam tikrą taisyklę kodus visoms operacijoms.

Konkrečiam uždaviniui spręsti galima nurodyti pagal tam tikra taisyklę suspaustu identifikatorių aibę, kuria nesunku (esant reikalui) išplėsti ir gauti originalų identifikatorių.

Pvz. 3. $< a_1 b_{1-3} d_{1,2} > \rightarrow < a_1 b_1 d_1, a_1 b_2 d_1, a_1 b_3 d_1, a_1 b_1 d_2, a_1 b_2 d_2, a_1 b_3 d_2 >$

Kadangi tokių identifikatorių eilutėje gali būti kiek norima daug ir jie gali būti išdėstyti reikiama tvarka, tai konkrečiam uždaviniui spręsti galime parinkti reikiamą identifikatorių kiekį, išdėstytu reikiama tvarka.

5.3. Nuosavos atributų reikšmių funkcinės savybės (f^{χ})

Nuosavos atributų reikšmių funkcinės savybės dažniausiai nurodo, kokiu algoritmu reikia apdoroti vieną ar keletą reliacinių aibių, kuriuose ryšys tarp atributų reikšmių yra nedidelis.

1) Pati paprasčiausia funkcinė savybė - tai algoritmas, kuris tikrina, ar duomo yra skaičius ar tekstas. Pateikiant duomenys apdorojimo sistemai yra svarbu nesuklysti. Akivaizdu, kad tekstas, patekės į aritmetinį apdorojimą rezultate duos klaidą.

2) Skaičiaus skaitinės reikšmės apibrėžimas. Ši funkcinė savybė užtikrina, jog skaičius, pateiktas apdorojimui, atitiks tam tikras sąlygas. Pvz.: $a \leq x \leq b$, $x \in N$, $x \notin N$, x > 2, ...

3) Apibrėžiamos visos skaitinės atributų reikšmės. Pvz.: $k \cdot a \leq x \leq k \cdot b$. Čia k - tai aibės galia (eilė, kortežų skaičius). Šios funkcinės savybės prasmė: visų atributu reikšmių vidurkis liks tam tikruose rėžiuose.

4) $f(c \vee \{c_i\})$. Tai reiškia, jog pateiktoji atributo reikšmė yra tapati vienam iš skaičių, kurie užrašyti pačioje funkcinėje savybėje. Pvz.: $c = 1$ arba 2 arba 3.

5) $f(c \vee \{j\})$ - reikšmė c yra tapati reikšmei, kuri egzistuoja tarp reliacinių aibių (kitos reliacinės aibės atributas, kurio domenas yra j). Pvz.: Duomenų bazėje įrašas yra įterpiamas, jeigu egzistuoja kitoje lentelėje. Reikšmės lauke, pažymėtame FOREIGN KEY įterpimas, jeigu egzistuoja atitinkama PRIMARY KEY reikšmė.

6) $f(c \vee \{j_1, j_2, \ldots, j_n\})$ - ši savybė yra panaši į penktą, tačiau į aibę įrašomos kitos reikšmės, iš rastos reikšmės kortežo. Pvz.: Duomenų bazėje surandama reikšmė (sakykime, pavardė), o įterpiama kita to kortežo reikšmė - raktas (PRIMARY KEY).

7) Atributo reikšmė kartojama tol, kol galutinis vartotojas pateikia naują reikšmę. Tada kartojama nauja reikšmė. Pvz.: Surenkant tekstą programa MS Word, vieną kartą nurodomas šriftas, ir jis bus naudojamas visam surinktam tekstui, kol nebus pasirinktas kitas šriftas.

8) $f(x)$, x nurodo dešimtainio taško vietą skaičiuje. (t.y. nustatomas tam tikras skaičiaus formatas). Pvz.: įvedant valiutą, visada galima įvesti tik 2 skaičius po kablelio.

9) Ši savybė yra panaši į aštunta. Skirtumas yra tame, jos įvesta reikšmė yra apvalinama iki tam tikro tikslumo. T.y. aštuntos savybės atvejų daugiau, negu du skaičius po kablelio neįmanoma įvesti, o šioje - įmanoma, tačiau skaičius bus suapvalintas.

10) Ši funkcinė savybė nurodo adresą aibėje r: $f(j)$, aibėje v: $f(i, j)$. Taip nurodoma kita funkcinė savybė, kuri yra kitoje funkcinėje aibėje. Pvz.: atlikus tam tikrą aritmetinę procedūrą, jos rezultatą užrašius į tam tikrą aibės-rezultato vietą, gali atsitikti taip, kad kita funkcinė savybė tą rezultatą perrašys. Kitas pavyzdys: atliekant tam tikros funkcinės savybės algoritmą, naudojamas kitos funkcinės savybės rezultatas, o pastaroji dar nebuvo įvykdyta.

11) "STOP" savybė: $f(\text{STOP}, i|j)$. Šios savybės prasmė: atėjus valdymui iki nurodytos funkcinės savybės adreso, visi darbai nutrūksta ir valdymas perduodamas galutiniam vartotojui.

12) "Kontrolinių sumų" savybė. (angliškai: "trailer"). Ši savybė yra naudojama duomenų tikslumui užtikrinti. Ji leidžia įsitikinti, kad perduodant duomenys, jie buvo gauti teisingai. Pvz:

					Eilutės sumos paskutinio skaičiaus papildinys iki 9
	1	3	4	5	6
	2	3	4	5	5
Stulpelio sumos paskutinio skaičiaus papildinys iki 9	6	3	1	9	

Kai duomenų kontrolinės sumos yra skaičiuojamos ir kortežams ir domenams (eilutėmis ir stulpeliais), tai galima nedalyvaujant žmogui ištaisyti vieną klaidą. Klaidos atstatymo algoritmas yra toks:

- Jeigu bent viename korteže kontrolinis skaičius nesutampa, tai ir vieno domeno kontrolinis skaičius irgi turi nesutapti. Tokiu būdu galima rasti neteisingo duomens adresą. Pagal skirtumus tarp esamų kontrolinių skaičių, galime išskaičiuoti teisingą duomens reikšmę.
- Jeigu kontrolinė suma nesutampa tik eilutėje arba tik stulpelyje - tai klaida yra kontrolinėje sumoje.

13) Skaičiaus kontrolė nuo iškraipymo. Įvedant duomenys visada egzistuoja klaidos tikimybė. Ši funkcinė savybė skiriasi nuo prieš tai buvusios tuo, kad skaičiuojant kontrolinė sumą yra įvertinamas ne tik skaitmenų rinkinys, bet ir jų pasirodymo tvarka. Pvz.: asmens kodas, ISBN/ISSN kodai. Šie kodai savyje turi vieną kontrolinį skaitmenį, kuris yra gautas sudėjus visus skaitmenys, prieš tai padauginus juos iš 2 laipsnyje (skaitmens pozicija skaičiuje).

14) Universalaus rūšiavimo reliacinės aibės viduje funkcinė savybė. Ši funkcinė savybė surūšiuoja kortežus pagal vieną iš domenų atributų reikšmes. Egzistuoja trys domeno atributų rūšiavimo būdai:

- didėjimo tvarka ($f(j <)$)
- mažėjimo tvarka ($f(j >)$)
- pagal šabloną ($f(jT)$) (location) pvz.: savaitės dienų pavadinimai

Ši atributų reikšmių funkcinių savybių aibė nėra uždara ir prireikus galima sukurti naujų funkcinių savybių bei realizuoti jų algoritmus programiniais moduliais. Kartais šioms savybėms yra įvedami pažymėjimai: $f^{x1}, f^{x2}, \ldots, f^{x14}, \ldots$.

5.4. Skaičiavimo-infologinės atributų reikšmių funkcinės savybės

Šios atributų reikšmių funkcinės savybės skirstomos i dvi tarpusavyje susijusios, panašias, tačiau atskiras grupes:

1) skaičiavimo funkcinės savybės f^{+}. Šios funkcinės savybės realizacija - tai aritmetinis skaičiavimas.

2) infologinės funkcinės savybės $f^{=}$. Šios funkcinės savybės realizavimui atliekami du skaičiavimai ir jų rezultatai palyginami (turi būti tenkinama viena iš palyginimo sąlygų: $=, <, >, \neq, \leq, \geq, \ldots$)

5.4.1. Skaičiavimo funkcinės savybės

Šios savybės pažymėjimas yra toks: $f^{+}(L\psi L\psi[c]\psi L\psi L\psi \ldots \rightarrow L)$. Čia L - atributo reikšmės adresas reliacinėje aibėje, ψ - aritmetine operacija, $[c]$ - tai konstanta, įrašoma į formulę vietoj atributo reikšmės, $\rightarrow$ - ženklas pažymi skaičiavimo pabaigą, L - tai vieta, kur turi būti patalpintas skaičiavimo rezultatas. Pvz.: $n_2 : n_1 \times [100] \rightarrow n_3$

A_1	A_2
n_1	n_2
n_3	n_4

Šių skaičiavimų esmė yra tokia: programiškai realizavus skaičiavimo algoritmą po to pakanka nurodyti tik skaičiavimo taisyklę.

5.4.2. Infologinės funkcinės savybės

Šios savybės pažymėjimas yra toks: $f^{=}(\underbrace{L\psi \ldots L\psi L}_{\alpha_1}\,\theta\,\underbrace{L\psi \ldots L\psi L}_{\alpha_2})$. Čia θ - tai operacijos ženklas ($=, <, >, \neq, \leq, \geq, \ldots$) Tai reiškia, kad apskaičiavus abu reiškinius (α_1 ir α_2), rezultatai yra palyginami. Ši funkcinė savybė leidžia patikrinti, ar operacijos rezultatai yra korektiški. Suprantama, kad vienoje skaičiavimo-infologinės funkcinės savybės išraiškoje ženklas θ gali būti nurodytas tik vieną kartą.

5.5. Skaičiavimo funkcinės savybės r-domenose

Atlikti aritmetinius veiksmus ir juos pateikti adresų eilutėmis galima tik todėl, kad r-kortežai yra tvarkiosios aibės. Jeigu schemų elementus būtų galima keisti, tai nagrinėjamoji funkcinė savybė neturėtų prasmės.

Ši funkcinė savybė įgalina realizuoti dviejų tipų algoritmus:

1) r-tipo reliacinėse aibėse atlikti skaičiavimus, gaunant įvairias tarpinės sumas.

2) duomenų laikmenose laikyti dalinai apdorotus duomenys.

Jeigu tam tikromis skaičiavimo sąlygomis duomenys yra stabilūs ir nereikalingos nuolatinės korekcijos, tada duomenų apdorojimas tampa lengvesnis.

Šios funkcinės savybės realizacijos pagrindą sudaro įvedamas į aibę r papildomas domenas kuriame nurodomi aritmetinių veiksmų algoritmai. Be to, įvedami požymių kodai, parodantys kuriuose domenuose turi būti atlikti skaičiavimai.

Pvz. 4.

$$\left[\begin{array}{c|cccc} & A_1 & A_2 & A_3 & A_4 \\ \hline \sigma_1 & 6 & 8 & 9 \\ \sigma_2 & 1 & 2 & 3 \\ \Sigma_1 & 10 & 9 & 0 \\ \sigma_1' & 4 & 1 & 6 \\ \sigma_2' & 3 & 2 & 1 \\ \Sigma_2 & 4 & 4 & 0 \\ \Sigma^2 & 10 & 13 & 0 \end{array}\right] ; \left(\begin{array}{ccccc} \Sigma_1 & = & \sigma_1 & + & \sigma_1' \\ \Sigma_2 & = & \sigma_2 & + & \sigma_2' \\ \Sigma^2 & = & \Sigma_1 & + & \Sigma_2' \end{array}\right)$$

Veiksmai atliekami virš domenų A_2 ir A_3. Domenas A_4 nesumuojamas (pvz.: A_4 yra tekstinio tipo).

Ši funkcinė savybė yra taikytina daugeliui buhalterinių bei kitų dokumentų, kurie yra išreiškiami reliacinėmis aibėmis ir reikalauja panašių algoritmų.

Jeigu tokie dokumentai keičiami retai - juos patogu laikyti tokia forma. Dažnai keičiantiems dokumentams reikės dažnai perskaičiuoti jų duomenys. Literatūroje ši savybė žymima taip: f_Σ.

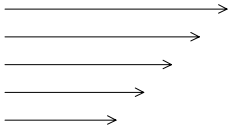
5.6. Programinių ėjimų funkcinės savybės

Ši funkcinė savybė yra žymima taip: $f^{\rightarrow}$ Atributų reikšmių aibėms r ir v atitinka funkcinių reikšmių aibės F_r ir f_v:

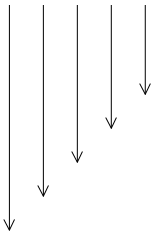
$$\begin{array}{ccccc} r & \leftrightarrow & F_r & = & < f, f, \ldots, f > \\ & & & & \left[\begin{array}{cccc} f & f & \ldots & f \\ f & f & \ldots & f \\ f & f & \ldots & f \\ \ldots & \ldots & \ldots & \ldots \\ f & f & \ldots & f \end{array}\right] \\ v & \leftrightarrow & F_v & = & \end{array}$$

Realizuojant funkcinės savybės, kyla klausymas: kokias iš jų reikia realizuoti anksčiau? Dažniausiai yra naudojami du reazilacijos būdai:

1) Didžioji dauguma funkcinių savybių (funkcinėje aibėje) realizuojamos iš kairės į dešinę ir po to iš viršaus į apačią.



2) Kita tvarka yra tokia: iš pradžių iš viršaus į apačią, po to iš kairės į dešinę.



Antrojo būdo realizacijos pavyzdys galį būti toks:

Pvz. 5. *Tegul turime tam tikrus lėšų sumas:*

Lėšos, skirtos organizacijai	*Lyginamasis kiekvienos institucijos svoris bendroje sumoje, procentais*
a	...
b	...
...	...
z	...

Norėdami surasti antro domeno atributų reikšmes, reikia iš pradžių suskaičiuoti bendrą sumą.

Nagrinėjant šią funkcinę savybę teorinių požiūrių, galime pastebėti tokius funkcinių savybių realizavimo eigos variantus:

$$F_v = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix}$$

1) $f^{\rightarrow} = f_1 f_2 f_3 f_4 f_5 f_6 f_7 f_8 f_9$

2) $f^{\rightarrow} = f_1 f_4 f_7 f_2 f_5 f_8 f_3 f_6 f_9$

3) $f^{\rightarrow} = f_3 f_2 f_1 f_6 f_5 f_4 f_9 f_8 f_7$

4) ...

Teorinių požiūrių funkcinėje aibėje ėjimai gali prasidėti bet kokiame elemente ir eiti bet kurią kryptimi.

Pvz. 6.

$$F_v + (2) =: F_v = \begin{bmatrix} 0 & 3 & 6 \\ 1 & 4 & 7 \\ 2 & 5 & 8 \end{bmatrix}$$

Šioje aibėje funkcinės savybės yra vykdomos skaičių didėjimo tvarka, nuo 0 iki 8.

Vienos funkcinės aibės, tos pačios schemos reliacinėms aibėms apdoroti dažniausiai pakanka tik duomenų įvedimui, jų korektiškumo patikrinimui ir, radus klaidas, korekcijos atlikimui. Realūs uždaviniai dažniausiai yra kompleksiniai, kuriems naudojamos įvairių schemų funkcinės aibės ir funkcinių aibių eilutės, apjungtos išorinių funkcinių savybių.

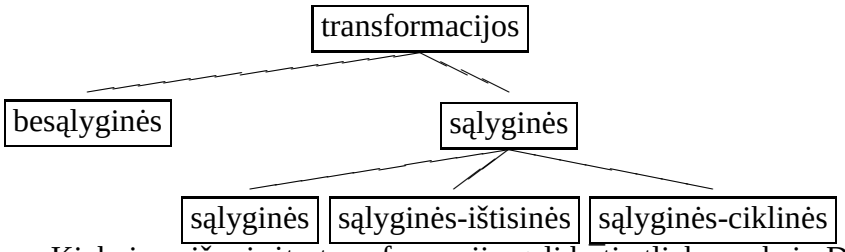
Išorinė funkcinė savybė nustato tarpusavio sąveika tarp dviejų eilutėje greta einančių funkcinių aibių.

5.7. Duomenų transformacijos funkcinės savybės

Duomenų transformacijos operacija įgalina iš vienos ar daugelio įvairių schemų aibių perkelti atributų reikšmes į vieną aibę. Tą aibę, į kurią įrašomos atributų reikšmes, vadinama *priimančiąją* aibe. Tos aibės, iš kurių imami duomenys, vadinamos *duomenų šaltinių*.

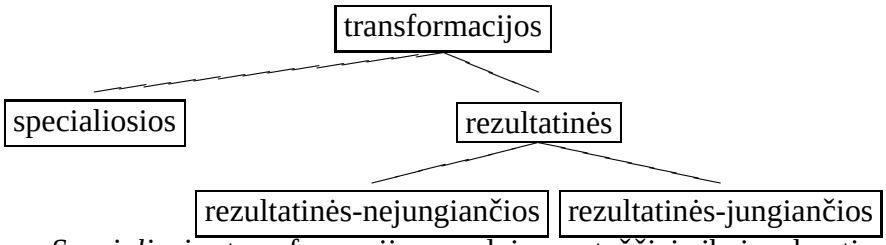
Priimančios aibės žymimos taip: r, v, u, N
Duomenų šaltiniai yra žymimi taip: r', v', u', $'N'$
Transformacija bendru atvejų paprastai žymima taip: $f^\sharp$. Konkreti transformacija yra žymima taip: $f^\sharp i$, kur $i \in N$.

Transformacijos skirstomos pagal įvairus kriterijus:



Kiekviena iš minėtų transformacijų gali būti atliekama kaip Dekarto transformacija, transformacija kortežuose ir transformacija domenose.

Atskiros transformacijų rūšys yra tokios:



Specialiosios transformacijos naudojamos tuščiai aibei sudaryti.

Transformacijas galima suskirstyti į tokias rūšys:

1) $r \leftarrow r'$ - transformacija atliekama iš kortežo į kortežą

2) $v \leftarrow v'$ - transformacija atliekama iš aibės į aibę

3) $r \leftarrow v'$ - transformacija atliekama iš aibės į kortežą

4) $v \leftarrow r'$ - transformacija atliekama iš kortežo į aibę

Bendriausias transformacijos formalus išreiškimas yra toks:
$f^\sharp(\nabla\{i|j\}^{n_1} < k|l >^{n_2} \overset{\theta}{\leftarrow} \triangle\{i|j\}^{n_3} < k|l >^{n_4})$
Čia n_1, n_2, n_3, n_4 - tai aibių, dalyvaujančių transformacijoje, kodai.

Vadinasi, transformacijoje vienu metu gali dalyvauti keturios aibės. Čia ženklas { } reiškia nurodytų aibių lyginimą naudojant θ sąlyga, o $<>$ - atributų reikšmių pernešimą. θ sąlyga gali būti viena iš: $=$, $\neq$, $<$,$>$, $\leq$, $\geq$ ir t.t.

Jeigu aibės n_3 adresų turinys, palyginimas su aibės n_1 adresų turiniu tenkina sąlyga θ, tai transformacija yra atliekama. Transformacijos atlikimas reiškia, jog adresų turinys iš aibės n_4 yra perneštas į aibę n_2 pagal nurodytus adresus.

Jeigu $n_1 \neq n_2 \neq n_3 \neq n_4$, tai transformacijoje dalyvauja keturios reliacinės aibės: dviejose aibėse (n_1 ir n_3) duomenys yra lyginami, o kitose - pernešami iš vienos į kitą (iš n_4 į n_2). Jeigu $n_1 \equiv n_2$; $n_3 \equiv n_4$, tai duomenys yra lyginami ir pernešami tose pačiuose reliacinėse aibėse. Tai pat galimi ir kitokie variantai.

Jeigu $n_1 \equiv n_2 \equiv n_3 \equiv n_4$, tai transformacija yra atliekama vienoje aibėje. Kitaip sakant, aibė yra reorganizuojama.

Ženklas ∇ reiškia priimantį reguliatorių, o $\triangle$ - duomenų išėmimo reguliatorių. Šie reguliatoriai yra natūriniai skaičiai, parodantys transformacijos žingsnį.

Transformacijos žingsnis reiškia, kiek kortežų reikia praleisti, atliekant transformaciją aibės r atžvilgių, kiek aibių praleisti v aibės atžvilgių. Jeigu priimančioji aibė yra v, tai priėmimo reguliatorius neturi prasmės.

Jeigu bendroje transformacijos formulėje n_1 ir n_3 adresų nėra, tai neturi prasmės ir sąlyga θ. Tuo atvejų transformacija vadinasi *besąlygine* ir žymima taip: $f^\sharp(\nabla < h|l > \leftarrow \triangle < h|l >)$.

Jeigu bendroje formulėje nėra aibės adresų n_2 ir n_4, tai transformacija vadinama rezultatine. Ši transformacija atliekama, bet duomenys nėra pernešami. Žymėjimas: $f^\sharp(\nabla\{i|j\}^{n_2} \overset{\theta}{\leftarrow} \triangle\{i|j\}^{n_4})$ Šios transformacijos prasmė yra tokia: palyginti įvairių aibių duomenys. Tai leidžia spręsti, ar duomenys yra korektiški.

Tvarkiosios aibės adresai turi griežtai nustatyta eilę.

Aibėje v adresai žymimi taip: $i|j$, $k|l$, o aibėje r pakanka šių adresų: j, l.

5.7.1. Besąlyginės transformacijos

Šios savybės pažymėjimas yra toks: $f^\sharp(\nabla < i|j >^{n_1} \leftarrow \triangle < k|l >^{n_2})$

Įmanomos tokios aibių n_1 ir n_2 kombinacijos: $n_1 \equiv n_2$ ir $n_1 \neq n_2$.

Pvz. 7. *Tarkime, yra tokie duomenys:* $u_1 = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$, $u_2 = \begin{bmatrix} c \\ d \\ e \end{bmatrix}$.

Šių aibių besąlyginė transformacija bus tokia (Dekarto transformacija):

$$\begin{bmatrix} a & c \\ a & d \\ a & e \\ b & c \\ b & d \\ b & e \\ c & c \\ c & d \\ c & e \end{bmatrix}$$

Tos pačios transformacijos rezultatas kortežu lygyje atrodys taip:

$$\begin{bmatrix} a & c \\ b & d \\ c & e \end{bmatrix}$$

O transformacijos domenų atžvilgių rezultatas yra toks (domenai neturi dubliuotis):

$$\begin{bmatrix} a \\ b \\ c \\ c \\ d \\ e \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

Įvertinant priėmimo ir išėjimo reguliatorius (∇ ir $\triangle$), transformacijos gali būti sudėtingesnės. Mes nagrinėsime paprasčiausios iš jų. Dažniausiai sutinkamos teorijoje ir praktikoje yra transformacijos korteže ir domene.

Pvz. 8. *Transformacija:* $r \leftarrow r'$*, duomenys:*

$$r = \left[\begin{array}{cc} A & B \\ \hline a & b \\ c & d \end{array}\right],\ r' = \left[\begin{array}{cc} C & D \\ \hline e & g \\ f & h \end{array}\right]$$

Transformacijos rezultatas, atitinkantis formulę $f^\sharp(<3|4><\leftarrow<1|2>)$ *bus toks:*

$$\left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & e & g \\ c & d & f & h \end{array}\right]$$

Šios transformacijos bendra idėja yra tokia: prie pradinės (priimančios) aibės pridedamos aibės-šaltinio atributų reikšmės pagal tam tikrą taisyklę.

Dekarto transformacijos rezultatas šiems duomenims bus toks:

$$\left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & e & g \\ a & b & f & h \\ c & d & e & g \\ c & d & f & h \end{array}\right]$$

Pvz. 9. v*-transformacija:* $v \leftarrow v'$*, duomenys:*

$$v = \left[\begin{array}{c|cc} & A & B \\ \hline C & a & b \\ D & c & d \end{array}\right],\ v' = \left[\begin{array}{c|cc} & A & B \\ \hline C & a_1 & b_1 \\ D & c_1 & d_1 \end{array}\right]$$

Transformacijos rezultatas, atitinkantis formulę $f^\sharp(<2|1,2|2><\leftarrow<2|1,2|2>)$ *bus toks:*

$$\left[\begin{array}{c|cc} & A & B \\ \hline C & a & b \\ D & c_1 & d_1 \end{array}\right]$$

Kaip jau minėjome, transformacijai $r \leftarrow v'$ reikia tik vieno priimančios aibės indekso.

Pvz. 10. *Transformacijos funkcija:* $f^\sharp(<1><\leftarrow<1|1>)$*, duomenys:*

$$r = \left[\begin{array}{cc} A & B \\ \hline a & b \\ c & d \end{array}\right],\ v' = \left[\begin{array}{c|cc} & C & D \\ \hline E & e & f \\ F & g & h \end{array}\right]$$

Transformacijos rezultatas:

$$\left[\begin{array}{cc} A & B \\ \hline e & b \\ c & d \end{array}\right]$$

Reliacinės aibės r-kortežas turi tam tikrą analogiją reliacinėje aibėje v.

Jeigu formulėje yra įrašyti reguliatoriai, tai transformacija tęsiasi, kol duomenų šaltinis neišsenka.

Pvz. 11. *Transformacijos formulė:* $f^\sharp(<1><\leftarrow<1|1>)$, $\nabla = 1$; $\triangle = 1$. *Pati transformacija:*

$$\left[\begin{array}{cc} A & B \\ \hline a & b \\ c & d \end{array}\right] \leftarrow \left[\begin{array}{c|cc} & C & D \\ \hline E & e & f \\ F & g & h \end{array}\right], \left[\begin{array}{c|cc} & C & D \\ \hline E & e_1 & f_1 \\ F & g_1 & h_1 \end{array}\right] = \left[\begin{array}{cc} A & B \\ \hline e & b \\ e_1 & d \end{array}\right]$$

∇ - *reguliatorius perstumia priimantį adresą žemyn:*

$$\left[\begin{array}{cc} A & B \\ \hline \boxed{a} & b \\ c & d \end{array}\right] \rightarrow \left[\begin{array}{cc} A & B \\ \hline a & b \\ \boxed{c} & d \end{array}\right]$$

$\triangle$ - *reguliatorius perstumia šaltinio adresą prie kitos sudėtinio šaltinio aibės:*

$$\left[\begin{array}{c|cc} & C & D \\ \hline E & \boxed{e} & f \\ F & g & h \end{array}\right], \left[\begin{array}{c|cc} & C & D \\ \hline E & e_1 & f_1 \\ F & g_1 & h_1 \end{array}\right] \rightarrow \left[\begin{array}{c|cc} & C & D \\ \hline E & e & f \\ F & g & h \end{array}\right], \left[\begin{array}{c|cc} & C & D \\ \hline E & \boxed{e_1} & f_1 \\ F & g_1 & h_1 \end{array}\right]$$

Taigi, galima iš daugelio aibių sugeneruoti reikiamo schemos aibę.

Dar vienas transformacijos variantas yra toks: $v \leftarrow r'$. Šis variantas yra grynai teorinio pobūdžio, sunkiai pritaikomas praktikoje. Transformacija atrodo taip:

$$[v] \leftarrow [r], [r], \ldots$$

Formule, atitinkančia šią transformacija yra tokia: $f^\sharp(\nabla < 1|1 > \leftarrow \triangle < 1 >)$. Priėmimo/išvedimo reguliatorius r aibėms slinks, o $[v]$ aibei liks toks pat. Tai reiškia, jog į galutinį rezultatą bus įrašyta paskutinės aibės iš aibių sekos $[r]$ reikšme.

Jeigu v aibė yra aibė-šaltinis, ji traktuojama kaip vienas kortežas. Transformaciją $[r] \leftarrow [v]$ atitinka formulė: $f^\sharp(\nabla < 1 > \leftarrow \triangle < 1|1 >)$.

5.7.2. Sąlyginės transformacijos

Sąlyginės transformacijos skirstomos į 3 grupes:

- sąlyginės
- sąlyginės-ištisinės
- sąlyginės-ciklinės

Sąlyginė transformacija vykdoma tol, kol nėra patenkinama tam tikra sąlyga. Toje duomenų šaltinio ir priimančios aibės vietoje, kur sąlyga tenkinama, transformacija atliekama tokiu pat būdu, kaip būtų atliekama besąlygine transformacija. Po pirmojo sąlygos patenkinimo bet kokie veiksmai nutrūksta.

Pirmasis trivialus sąlyginės transformacijos variantas yra toks, kad esant šiai transformacijai duomenys iš šaltinio į priimančią aibę nepernešami.

Bendra šios funkcinės savybės formulė užrašoma taip: $f^\sharp(\nabla\{i|j\}^{n_1} \overset{\theta}{\leftarrow} \triangle\{k|l\}^{n_2})$. Čia taip pat galimi du variantai: $n_1 \equiv n_2$ ir $n_1 \neq n_2$. θ - tai palyginimo operatorius, ir gali būti vienas iš: $=$, $\neq$, $<$,$>$, $\leq$, $\geq$ ir t.t.

Svarbu prisiminti, kad palyginamas yra adresų turinys, o ne patys adresai.

$\{\ \}$ - tai adresų aibė, kuri bendru atvejų nėra tvarkioji: $\{1,2,3\} = \{3,2,1\}$, palyginkite su: $<1,2,3> \neq <3,2,1>$.

Tegul turime dvi aibės:

$$r = \left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & c & f \\ d & e & a & c \\ a & b & c & d \end{array}\right] \text{ ir } r_1 = \left[\begin{array}{ccc} C & B & D \\ \hline a_1 & a_1 & a_1 \\ a & e & c \\ b_1 & b_2 & b_3 \end{array}\right]$$

Tegul transformacijos formulė yra tokia: $f^\sharp(\nabla\{1\}^r < 2 >^r \overset{=}{\leftarrow} \triangle\{1\}^{r_1} < 1 >^{r_1})$

Šios operacijos rezultatas yra toks:

$$\left[\begin{array}{cccc} A & B & C & D \\ \hline a & \boxed{\text{e}} & c & f \\ d & e & a & c \\ a & b & c & d \end{array}\right]$$

Kadangi čia buvo *sąlyginė* transformacija, tai transformavus duomenys vieną kartą, aibės toliau nebelyginamos.

Šios funkcinės savybės taikymo pavyzdys: tarkime, reikia surasti studentų grupių aibėje konkretaus studento pavardę. Suradus reikiamą pavardę, transformacija (t.y. paieška) baigiasi.

Jeigu reikia rasti sąrašuose tam tikros pavardės bendrapavardžius, transformaciją reikia tęsti, kol išseks duomenų šaltinis. Šiuo atvejų transformacija vadinama *sąlygine-ištisine*.

Jeigu vykdant sąlyginę-ištisinę transformaciją duomenų šaltinis išsenka, ir reikia grįžti prie aibės pradžios, tęsiant vykdymą, tai transformacija vadinasi *sąlyginė-ciklinė*. Tokia transformacija turi prasmę tik tada, kai pakeičiamas pirmosios lyginamos aibės turinys.

Sąlyginė transformacija. Šios transformacijos vienas iš atvejų: $r \leftarrow r'$. Bendru atvejų šios transformacijos šaltinis, susidedantis iš daugelio aibių atrodo taip:

$$\left[\begin{array}{ccccc} A & B & C & D & \dots \\ \hline a & \dots & \dots & \dots & \dots \\ b & \dots & \dots & \dots & \dots \\ c & \dots & \dots & \dots & \dots \\ d & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{array}\right], \dots$$

Pvz. 12. *Tarkime, aibė su kuria atliksime* sąlygine *transformacija yra tokia:*

$$r = \left[\begin{array}{cccc} A & B & C & D \\ \hline a & 1 & b & 2 \end{array}\right]$$

O aibė-šaltinis yra tokia:

$$r_1 = \left[\begin{array}{ccc} A & B & C \\ \hline a & c & 6 \\ b & d & a \\ c & 1 & b \end{array}\right]$$

Transformacijos sąlyga yra tokia: $\{1, 2, 4\} \overset{\cup}{\leftarrow} \{3\}$. *Tada šios transformacijos rezultatas:* $< b, d, a >$. *(Ši sąlyga reiškia, tikrinimą, ar aibės-šaltinio* C *domeno reikšmės yra lygios* r *aibės* A, B *arba* D *domeno reikšmėms.)*

Keičiant transformacijos sąlyga į tokią: $\{1, 2, 4\} \overset{\sqsupset}{\leftarrow} \{3\}$, *rezultatas bus:* $< a, c, 6 >$. *(*r *aibės domenose (*A, B, D*) nėra reikšmės iš* r_1 *aibės* C *domeno.)*

Sąlyginė-rezultatinė transformacija. Ši transformacija jungia ne tik duomenų palyginimą ir/arba pernešimą, bet ir aritmetinės operacijos su tais duomenimis (pvz.: atimtis, sudėtis, ...).

Pvz. 13. *Tarkime, turime tokius pradinius duomenys:*

$$\left[\begin{array}{ccc} A & B & C \\ \hline a & 1 & 2 \\ b & 3 & 4 \\ c & 5 & 6 \end{array}\right], \left[\begin{array}{ccc} A & B & C \\ \hline a & 2 & 3 \\ d & 1 & 8 \\ e & 1 & 6 \end{array}\right]$$

ir bendrą transformacijos formulę: $f^\sharp(\{1\} < 2,3 >\overleftarrow{=} \{1\} < 2,3 >)$

Šios transformacijos rezultatas galėtų būti toks (nejungiančia transformacija, kurios rezultatas - atitinkamų kortežų sumą):

$$\left[\begin{array}{ccc} A & B & C \\ \hline a & 3 & 5 \\ b & 3 & 4 \\ c & 5 & 6 \end{array}\right]$$

Jeigu transformacija yra sumojančia ir jungiančia, tai rezultatas bus kitoks:

$$\left[\begin{array}{ccc} A & B & C \\ \hline a & 3 & 5 \\ b & 3 & 4 \\ c & 5 & 6 \\ d & 1 & 8 \\ e & 1 & 6 \end{array}\right]$$

Šios transformacija naudojimo pavyzdys: sandelio informacinė sistema. Pateikto pavyzdžio atvejis - tai prekių atvežimas į sandelį: (prie esamų prekių pridėti naujai atvežtų prekių kiekius ir tiesiog pridėti prekes, kuriu dar nėra sandelį).

Sąlyginė-ištisinė transformacija

$$\left[\left[\begin{array}{cc} A & B \\ \hline a & b \\ c & d \\ e & f \end{array}\right]^1 \left[\begin{array}{cc} C & D \\ \hline \emptyset & \emptyset \end{array}\right]^2\right]^N \overleftarrow{=} \left[\left[\begin{array}{cccc} A & B & C & D \\ \hline f & a & a & b \\ c & g & c & g \\ c & d & 1 & 2 \end{array}\right], \left[\begin{array}{cccc} A & B & C & D \\ \hline e & f & 3 & 4 \\ d & f & 1 & 6 \end{array}\right], \left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & 5 & 6 \\ a & b & 3 & 3 \\ c & d & 4 & 4 \end{array}\right]\right]^3$$

Transformacijos formulė: $f^\sharp(\nabla\{1,2\}^1 < 1,2 >^2 \overleftarrow{=} \triangle\{1,2\}^3 < 3,4 >^3)$, $\nabla = 1$, $\triangle = 1$. Transformavus pagal šią formulę, gausime:

$$\left[\begin{array}{cc} C & D \\ \hline 5 & 6 \\ 1 & 2 \\ 3 & 4 \end{array}\right]$$

Sąlyginė-ciklinė transformacija. Tarkime, transformacijos formulė yra tokia:

$$f^\sharp(\{1\} < 1,3,4 >\overleftarrow{=} \{1\} < 1,2,3 >)$$

, o duomenys:

$$\left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & \emptyset & \emptyset \\ c & d & \emptyset & \emptyset \\ e & f & \emptyset & \emptyset \end{array}\right] \overleftarrow{=} \left[\left[\begin{array}{ccc} A & C & D \\ \hline e & d & d \\ f & a & b \\ c & c & d \end{array}\right], \left[\begin{array}{ccc} A & C & D \\ \hline a & a & b \\ c & c & d \\ e & c & d \end{array}\right], \left[\begin{array}{ccc} A & C & D \\ \hline e & c & d \\ e & f & g \\ a & d & c \end{array}\right]\right]$$

Ciklinėse transformacijose priėmimo/išėmimo reguliatoriai veikia automatiškai.

Šios transformacijos rezultatas yra toks:

$$\left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & a & b \\ a & b & d & c \\ c & d & c & d \\ c & d & c & d \\ e & f & d & d \\ e & f & c & d \\ e & f & c & d \\ e & f & f & g \end{array}\right] =^{\text{išmetami dublikatai}} \left[\begin{array}{cccc} A & B & C & D \\ \hline a & b & a & b \\ a & b & d & c \\ c & d & c & d \\ e & f & d & d \\ e & f & c & d \\ e & f & f & g \end{array}\right]$$

5.8. Išorinės funkcinės savybės

Funkcinė aibė skirta apdoroti duomenims arba atributų reikšmėms, kurios yra tos pačios schemos reliacinės aibės egzemplioriai.

$$F_u \leftrightarrow < u >$$

Tai dažniausiai būna patikrinimas ar pateikiami duomenys yra teisingi, klaidų taisymas, duomenų skirtinguose aibėse palyginimas ir t.t.

Sprendžiant realų uždavinį, nepakanka vienos funkcinės savybės ir vienos schemos aibės. Taigi, susiduriame su aibių aibe, funkcinėmis aibėmis ir dažniausiai dideliu kiekiu įvairių schemų reliacinėmis aibėmis. Kyla klausymas: kaip ir kokia tvarka naudoti funkcinės aibės, ir kokios yra šių aibių tarpusavio sąsajos? Atsakymui į šiuos klausymus naudojamos *funkcinių aibių eilutės*. Dvi eilutėje gretimos funkcinės aibės susiejamos išorine funkcine savybe: ψ.

Taigi, praktikoje dažniausiai yra naudojamos aibių eilutės:

- Apdorojamų aibių eilutė: $<< u_a >< u_b >< u_c > \ldots < u_z >>$
- Funkcinių aibių eilutė (apdorojančioji eilutė): $< F_{u_1} \psi F_{u_2} \psi \ldots \psi F_{u_n} >$

Taigi, išorinė funkcinė savybė nurodo, ar eilutėje einamąją aibe reikia pakeisti kita (analizuojant reliacinės aibės), ar ne. Jeigu keisti nereikia, funkcinė aibė žymima taip: $\bar{F_u}$, jeigu reikia keisti: $\tilde{F_u}$.

Ta pati funkcinė savybė ψ nurodo, ar apdorojamąją aibę reikia keisti kita: $\bar{u}$ - jeigu reikia keisti, $\tilde{u}$ - jeigu nereikia.

Šiuo atveju realizuojant F_u ir u apdorojimą, būtina turėti trys iš anksto nustatytos sritys (S_i):

- S_1 - tai sritis, kurioje nagrinėjama F_u
- S_2 - tai sritis, kurioje nagrinėjama einamoji u
- S_3 - tai tranzitinė sritis

Jeigu žymime, kad u pernešama, tai pernešimas vyksta iš S_2 į S_3: $u \rightarrow: S_2 \rightarrow S_3$, $u \downarrow: \rightarrow S_3$

Tranzitinė sritis reikalinga tarpinių duomenų apdorojimo rezultatų apjungimui i vieną (tarpinę) aibę.

Išorinės funkcinės savybės ψ variantai:

	S_1	S_2	S_3	Aprašymas
ψ^1	$\tilde{F}$	$\bar{u}$	$u\downarrow$	Ši funkcinė savybė reiškia, jog duomenys, esantys aibėje u reikalauja sudėtingo apdorojimo, todėl iš funkcinių aibių eilutės imama nauja funkcinė aibė, apdorojimas tęsiamas, o po to duomenys pernešami į tranzininę sritį
ψ^2	$\tilde{F}$	$\bar{u}$	_	Duomenys nepernešami, lieka tokie pat
ψ^3	$\tilde{F}$	$\tilde{u}\rightarrow$	u	Keičiasi ir F ir u, prieš tai įnešant u į tranzitinę sritį
ψ^4	$\tilde{F}$	$\bar{u}\rightarrow$	u	Padaroma u aibės kopija tranzitinėje srityje
ψ^5	$\bar{F}$	$\bar{u}$	$u\downarrow$	Funkcijos ir duomenys nekeičiami. Duomenys pernešami į S_2
ψ^6		$\bar{u}$		Duomenys išsaugomi naujiems uždaviniams
ψ^7		$\bar{u}$		Duomenys išsaugomi naujiems uždaviniams ir pateikiami vartotojui
ψ^8		$\bar{u}$		Duomenys pateikiami vartotojui

5.9. Pagalbinės funkcinės savybės

Šios funkcinės savybės leidžia sukurti struktūrą (t.y. aibę), turinčia tam tikras savybės (matmenys, domenus) ir tuščias atributų reikšmes.

6. Matematinis taikomojo uždavinio modelis

Matematinį taikomojo uždavinio modelį sudaro identifikatoriai, kurie išreiškiami tokia formule:

$$dom(\omega_1, \omega_2, \ldots, \omega_n)$$

$$dom(\omega_1), dom(\omega_2), \ldots, dom(\omega_n)$$

Svarbiausia struktūra yra $a_i b_j d_k < c_{ij} >$, kuri reiškia, duomenų aibę aprašančia realaus pasaulio objektą, subjektą, reiškinį ir panašiai.

Šį aibė būtinai turi būti sujungta: $a_i \rightarrow < R_i >$. R_i - tai aibių schema.

Jeigu $a_{ij} \rightarrow v$, tai būtinai turi egzistuoti $F_{a_{ij}}$. Tai yra dėl to, kad galima būtų susieti: $F_u \leftrightarrow C_{ij} = \{a_{ij}\}$.

$$\bar{F} = \begin{Bmatrix} f & f & f & \ldots \\ f & f & f & \ldots \\ f & f & f & \ldots \\ \ldots & \ldots & \ldots & \ldots \end{Bmatrix}$$

Čia $\bar{f}$ - visų funkcinių savybių aibė. Konkreti funkcinių savybių aibė - tai aibės $\bar{F}$ poaibis: $F_u \subset \bar{F}$

Vienas iš pagrindinių adaptyvumo principų: jeigu atsitinka taip, kad apdoroti konkrečią aibę u arba N, aibėje $\bar{F}$ neatsiranda reikiamos funkcinės savybės, tai adaptyvioji sistema leidžia sukurti naują funkcine savybe, apimančia operacijas, kurių iki šiol negalėjome atlikti, ir prideda tą funkcine savybe prie aibės $\bar{F}$.

Adaptyvumas pasireiškia tuo, kad funkcinių savybių aibė nagrinėjama ne kaip nestandartinis modulis, o kaip $\bar{F}$ aibės narys. Tokių atvejų atliekama ne tik ta reikalinga operacija, bet ir nauja funkcinė savybė kombinuojama su buvusiomis funkcinėmis savybėmis, išplečia sistemos funkcinės galimybės.

Funkcinių savybių aibei $\bar{F}$ didėjant, šiuo atvejų - naujų funkcinių savybių reikšmė sparčiai mažėja. (t.y. mažėja tikimybė, kad reikiamos funkcinės savybės nėra aibėje $\bar{F}$).

Taigi, antrasis pagrindinis adaptacijos principas yra toks: *bet kokio uždavinio sprendimas prideda prie aibės $\bar{F}$ jam vienam charakteringa ir vis kitokia esamų funkcinių savybių realizaciją.*

6.1. Duomenų pateikimo modeliavimas

Duomenys bendroje formoje galima užrašyti taip:

$$M = a_i f_{B_j} f_{C_k} F$$

Tai reiškia, kad tokių būdų gali būti aprašyti visi duomenys su savo esamomis funkcinėmis savybėmis. Čia a_i - duomenų struktūrų (schemų) identifikatoriai (viso schemų yra i), f_{B_j} - subjekto identifikatoriai (j - tai subjektų kiekis), f_{C_k} - laiko faktorių identifikatoriai (k - tai laiko faktorių kiekis).

Subjekto ir laiko faktoriaus identifikatoriams gali būti priskirtos tam tikros funkcinės savybės, todėl tokios ir yra jų išraiškos.

Pvz. 14. *Transporto įmonės (kaip objekto) vieni iš subjektų - tai automobiliai.*

Identifikatorių egzistavimas nereiškia identifikuojamo objekto egzistavimo: identifikatorius gali nurodyti neegzistuojantį objektą.

Duomenų struktūros identifikatoriaus neužtenka vienareikšmiškam objekto identifikavimui, todėl šiam identifikatoriui nepriskiriama funkcinės savybės.

Pagal kiekvieną duomenų struktūrą toliau seka duomenų aibių sistemos paieškos identifikatorius: $R(\omega N)$. Čia ω - paieškos aibės identifikatorius.

Pagal kiekvieną duomenų struktūrą yra pateikiamas paieškos identifikatorius (ω), aibės schema (a_i) bei patys duomenys: $\{a_i b_j c_k N\}$. Identifikatorius ω nėra tiesiogiai susietas su duomenų išdėstymų.

Pvz. 15.

$$U = \left[\begin{array}{ccc} \dots & A & \dots \\ \hline \dots & \dots & \dots \\ \dots & C & \dots \\ \dots & \dots & \dots \end{array}\right]$$

$C = \{f_1, \dots, f_i, \dots, f_n\}$, $f_i \in F$

Atributo reikšmė C susieta su tam tikroms funkcinėms savybėms: $f_1, f_2, \dots, f_n$, kurios visos priklauso funkcinių savybių aibei F, kuri yra priskirta visiems u aibės atributams.

Čia nėra pavaizduotos funkcinės savybės susijusios su daugiau negu viena reliacinės aibės atributo reikšme.

Apibendrintas i-tojo elementaraus uždavinio matematinis modelis gali būti užrašomas taip:

$$Z_i = \mathrm{dom}(\omega)(U_i \xleftarrow{<F>} < N' >)$$

Šitie modeliai yra naudotini, kai yra daug duomenų ir palyginus nedaug (matematinių) operacijų.

6.2. Bendrasis uždavinio matematinis modelis

Uždavinio rezultatas yra toks:

$$Z_i =< N_m^d >$$

Čia $< N_m^d >$ - tai aibių sistema: $< N_m^d >=<< F_{k_1}^d > \psi < F_{k_2}^d > \psi \dots \psi < F_{k_n}^d >> \in Z$, $\forall F_{k_i}^d, \psi \in N'$.

Bendras uždavinys atrodo taip:

$$Z = \begin{array}{c} \mathrm{dom}(\omega_1)\big(U_1 \xleftarrow{<F_1>} < N_1' > \big) \\ \mathrm{dom}(\omega_2)\big(U_2 \xleftarrow{<F_2>} < N_2' > \big) \\ \dots \\ \mathrm{dom}(\omega_t)\big(U_t \xleftarrow{<F_t>} < N_t' > \big) \end{array}$$

$$< F_i >=< F\psi F\psi \ldots \psi F\psi F >$$
$$< N >=< U_1, U_2, \ldots, U_n >$$
$$< Z_i >\equiv< N >$$

7. Operacijos ADD, DEL ir CHANGE

Taip pat išskiriamos trys operacijos skirtos aibių apdorojimui:

- operacija ADD
- operacija DEL
- operacija CHANGE

7.1. ADD operacija

Ši operacija prideda (sukuria naują) reliacinę aibė. Tai yra CREATE TABLE operatoriaus ekvivalentas SQL kalboje.

7.2. DEL operacija

Ši operacija ištrina reliacinę aibę. Aibės pašalinimas gali būti suprastas r tipo aibėms kaip visos aibės pašalinimas, o v tipo aibėms - kaip visos aibės arba jos dalies pašalinimas. Tai yra DROP TABLE operatoriaus ekvivalentas SQL kalboje.

7.3. CHANGE operacija

Ši operacija pakeičia reliacinę aibę. Tai yra ALTER TABLE operatoriaus ekvivalentas SQL kalboje.