Capitolul 3

DEPENDENŢE FUNCŢIONALE

Proiectarea logică a bazei de date urmăreşte printre altele diminuarea redundanţei şi asigurarea securităţii datelor. Acest scop se poate atinge, dacă se cunosc a priori constrângerile ce pot fi aplicate asupra datelor. Dependenţele sunt constrângeri impuse datelor în baza de date. Ba mai mult, mulţimea de dependenţe este partea esenţială a schemei unei relaţii, deci şi a schemei bazei de date. Dependenţele funcţionale au fost primele constrângeri logice considerate în modelul relaţional. Ele formează cel mai simplu şi cel mai larg răspândit tip de dependenţe.

Prezentul capitol e consacrat regulilor de inferenţă, închiderilor şi diverselor forme de acoperiri ale dependenţelor funcţionale.

3.1. Noţiuni generale

Să considerăm relaţia *orar* din fig. 3.1.

orar	PROFESOR	DISCIPLINĂ	ZI	ORĂ	GRUPĂ	SALĂ
	Petrescu	Baze de date	Luni	8:00	C941	402
	Petrescu	Baze de date	Mierc.	14:30	C941	216
	Petrescu	Baze de date	Mierc.	16:00	C941	216
	Vasilache	Progr.logică	Luni	9:30	C941	404

Fig.3.1. Relaţia *orar*

Această relaţie arată care profesor predă disciplina dată, cărei grupe, în ce zi a săptămânii, la ce oră şi în ce sală. Atributele ce formează schema acestei relaţii nu pot primi orice valori. Atributele se află într-o interdependenţă. Aici, în particular, se suprapun asupra atributelor următoarele constrângeri:

(1) o disciplină este predată unei grupe de studiu de un singur profesor;
(2) profesorul, în ziua dată, la ora dată se găseşte într-o singură sală;
(3) în ziua dată, la ora dată, în sala dată se predă o singură disciplină.

Aceste constrângeri ce reflectă o interdependenţă între atribute sunt exemple de dependenţe funcţionale. Dependenţa funcţională este o generalizare a noţiunii de cheie.

Constrângerile de mai sus pot fi formulate:

(1) DISCIPLINĂ GRUPĂ determină funcţional PROFESOR sau, ce e echivalent PROFESOR e determinat funcţional de DISCIPLINĂ GRUPĂ;
(2) PROFESOR ZI ORĂ determină funcţional SALĂ;
(3) ZI ORĂ SALĂ determină funcţional DISCIPLINĂ;

şi notate respectiv:

(1) DISCIPLINĂ GRUPĂ $\rightarrow$ PROFESOR;
(2) PROFESOR ZI ORA $\rightarrow$ SALA;
(3) ZI ORA SALĂ $\rightarrow$ DISCIPLINA.

Unica posibilitate de a determina dependențele funcționale constă într-o analiză cu luare-aminte a semanticii atributelor. În acest sens dependențele sunt de fapt aserțiuni despre lumea reală. Ele nu pot fi demonstrate. Dar ele pot şi trebuie să fie susținute de SGBD-uri. Majoritatea sistemelor susțin numai dependențele funcționale determinate de cheile relației. Dar sunt şi sisteme ce susțin dependențe funcționale arbitrare.

Trebuie menționat că declararea dependențelor funcționale într-o bază de date este o decizie pe care o ia numai proiectantul bazei de date. Odată declarate SGBD-ul va susține aceste constrângeri. În afară de aceasta, după cum se va vedea în celelalte secțiuni, grație dependențelor, există o structură mai eficientă de păstrare a datelor. Dependențele funcționale vor servi la proiectarea schemelor bazelor de date cu anumite proprietăți dezirabile.

Definiția 3.1. Fie relația r cu schema R şi $X,Y \subseteq R$. Vom spune că dependența funcțională $X \rightarrow Y$ este validă în relația r (sau relația r satisface dependența funcțională $X \rightarrow Y$), dacă, pentru orice două tupluri din r, fie t_1 şi t_2, din condiția că tuplurile au X-valori identice, urmează că au şi Y-valori identice, adică $t_1[X]=t_2[X] \Rightarrow t_1[Y]=t_2[Y]$.

Dacă $X \rightarrow Y$ e validă în r(R), vom spune că X *determină funcțional* Y sau, că Y e *determinat funcțional* de X. În această definiție (şi mai departe) simbolul "$\Rightarrow$" notează "implică".

Deci dependența funcțională $X \rightarrow Y$ reprezintă o restricție de integritate aplicată tuplurilor relației r(R), în sensul că oricare două tupluri din r care prezintă o aceeaşi valoare pentru X trebuie să prezinte o aceeaşi valoare pentru Y.

Definiția 3.1 poate fi interpretată şi în felul următor: relația r(R) satisface dependența funcțională $X \rightarrow Y$, dacă relația $\pi_Y(\sigma_{X=x}(r))$ conține nu mai mult de un tuplu pentru orice valoare x a atributului X.

Partea stângă a dependenței poartă numele de *determinant*, iar partea dreaptă a dependenței poartă numele de *determinat*. Astfel în cadrul dependenței $X \rightarrow Y$, X este determinantul, iar Y determinatul.

Exemplul 3.1. Considerăm relațiile din fig.3.2. În ele sunt valide următoarele dependențe funcționale. În relația r_1: $A \rightarrow B$; în relația r_2: $A \rightarrow B$, $B \rightarrow A$; în relația r_3: $A \rightarrow B$.

r_1	A	B
	a_1	b_1
	a_2	b_2
	a_3	b_1
	a_4	b_1
	a_5	b_2
	a_6	b_2

r_2	A	B
	a_1	b_1
	a_2	b_4
	a_1	b_1
	a_3	b_2
	a_2	b_4
	a_4	b_3

r_3	A	B
	a_1	b_1
	a_2	b_4
	a_1	b_1
	a_3	b_2
	a_2	b_4
	a_4	b_4

Fig.3.2. Relațiile r_1, r_2 şi r_3.

Pentru a verifica dacă o dependență e validă într-o relație dată, se utilizează următorul algoritm.

Algoritmul SATISF (r, $X \rightarrow Y$)

Intrare: Relația r(R), dependența funcțională $X \rightarrow Y$, unde $X,Y \subseteq R$.

Ieşire: *Adevăr*, dacă relația r satisface dependența funcțională $X \rightarrow Y$; *fals* – în caz contrar.

(1) Se sortează tuplurile relației r în aşa fel, ca tuplurile cu X-valori egale să fie grupate împreună.

(2) Se verifică dacă mulțimea de tupluri cu X-valori egale are şi Y-valori egale, atunci la ieşire obținem adevăr; în caz contrar - fals.

Menționăm că nu ne interesează dependențele funcționale întâmplătoare, dar numai acele ce decurg din semantica atributelor. De exemplu, în relația *orar* e validă şi dependența funcțională PROFESOR→DISCIPLINĂ. Dar ea nu reprezintă o constrângere ce reflectă lumea reală, fiindcă în realitate un profesor poate şi, de regulă, predă mai multe discipline.

Numai dependențele neîntâmplătoare, asigură integritatea semantică a bazei de date. De exemplu, dacă un utilizator doreşte să insereze în relația *orar* un tuplu:

Add(*orar*; <Vasilache "Structuri de date" luni 8:00 c942 402>),

sistemul de gestiune va stopa efectuarea acestei operații, fiindcă va fi violată dependența funcțională ZI ORĂ SALĂ → DISCIPLINĂ. Dacă SGBD-ul nu susține dependențele funcționale, atunci se poate întâmpla că într-o sală în acelaşi timp se vor preda două discipline diferite.

3.2. Reguli de inferență

Într-o relație r(R) în orice moment sunt valide o mulțime de dependențe funcționale, să zicem F. Adică F este o mulțime de dependențe satisfăcută de relația r(R). Aici apare aceeaşi problemă ca şi în cazul cheilor. O extensie a relației satisface mulțimea F de dependențe funcționale, în timp ce altă extensie nu satisface. Pe noi ne interesează numai dependențele ce sunt satisfăcute de orice extensie a relației r(R). Dar pentru aceasta sunt necesare cunoştințe asupra semanticii relației r(R). Vom considera că mulțimea F de dependențe funcționale este definită pe mulțimea R de atribute ce formează schema relației r şi orice extensie a relației r satisface mulțimea F.

Evident că mulțimea de dependențe valide în r(R) este finită, fiindcă finită este schema R. Prin urmare, s-ar putea verifica dacă r satisface dependențele din F, aplicând algoritmul SATISF. Însă astfel de soluție este foarte laborioasă. Există o altă cale. Dacă sunt cunoscute nişte dependențe, din ele pot fi deduse altele.

Definiția 3.2. Fie relația r cu schema R, F o mulțime de dependențe funcționale şi f o dependență asupra R. Notăm cu SAT(F) mulțimea tuturor relațiilor asupra R ce satisface orice dependență din F. Vom spune că F *logic implică* f, sau f este *consecință logică* a F, scriem $F|=f$, dacă orice relație r(R) ce satisface dependențele din F satisface şi f, adică

$$r(R) \in SAT(F) \Rightarrow r(R) \in SAT(F \cup \{f\}).$$

O regulă de inferență stabileşte că, dacă o relație satisface anumite dependențe, ea trebuie să satisfacă şi alte dependențe. Vom considera şase reguli de inferență a dependențelor funcționale.

Fie relația r definită pe mulțimea de atribute R şi fie $W,X,Y,Z \subseteq R$.

DF1. Regula reflexivității. Dacă $Y \subseteq X$, atunci $X \rightarrow Y$.

Demonstrarea acestei afirmații este evidentă. Nu putem avea într-o relație două tupluri cu X-valori egale şi să nu fie egale componentele lor pentru o submulțime a lui X.

Definiția 3.3. O dependență $X \rightarrow Y$ este *trivială* dacă $Y \subseteq X$.

Regula DF1 generează numai dependențe triviale şi ea nu depinde de F. Întrucât $\emptyset \subseteq X \subseteq R$, atunci $X \rightarrow \emptyset$ şi $R \rightarrow X$ sunt dependențe triviale. Deoarece $X \subseteq X$, $X \rightarrow X$ e dependență trivială. Dintre aceste dependențe prima, $X \rightarrow \emptyset$, nu are nici o aplicare practică.

DF2. Regula incrementării. Dacă $X \rightarrow Y$ şi $Z \subseteq W$, atunci $XW \rightarrow YZ$.

Demonstrație. Fie r satisface dependența funcțională $X \rightarrow Y$, însă în ea există două tupluri t_1 şi t_2 cu XW-valori egale, dar componentele YZ nu coincid. Conform regulii DF1 tuplurile t_1 şi t_2 au Z-valori egale (fiindcă $Z \subseteq W$). Atunci tuplurile trebuie să nu coincidă măcar pe un atribut din Y. Dar aceasta înseamnă că tuplurile t_1 şi t_2 au X-valori egale şi nu au Y-valori egale, ce contrazice ipoteza că relația r satisface dependența funcțională $X \rightarrow Y$.

Să observăm că regula DF2 are mai multe cazuri speciale. Dacă $Z=\emptyset$ şi $X \rightarrow Y$, atunci $XW \rightarrow Y$ pentru orice submulțime W din R. Dacă W=Z şi $X \rightarrow Y$, atunci $XW \rightarrow YW$. Dacă X=W, Z=X şi $X \rightarrow Y$, atunci $XX \rightarrow XY$, adică $X \rightarrow XY$.

r	A	B	C	D
	a_1	b_1	c_1	d_1
	a_2	b_2	c_1	d_1
	a_1	b_1	c_1	d_2
	a_3	b_3	c_2	d_3

Fig.3.3.

Exemplul 3.2. Considerăm relația din figura 3.3. Relația r(ABCD) satisface dependența funcțională $A \rightarrow B$. Conform regulii DF2 în această relație sunt valide şi dependențele $AB \rightarrow B$, $AC \rightarrow B$, $AD \rightarrow B$, $ABC \rightarrow B$, $ABD \rightarrow B$, $ACD \rightarrow B$, $ABCD \rightarrow B$, $AC \rightarrow BC$, $AD \rightarrow BD$, $ABC \rightarrow BC$, $ABD \rightarrow BD$, $ACD \rightarrow BC$, $ACD \rightarrow BD$, $ACD \rightarrow BCD$, $ABCD \rightarrow BC$, $ABCD \rightarrow BD$, $ABCD \rightarrow BCD$.

DF3. Regula aditivității. Dacă $X \rightarrow Y$ şi $X \rightarrow Z$, atunci $X \rightarrow YZ$.

Demonstrație. Presupunem contrariul: în relația r(R) există două tupluri t_1 şi t_2, pentru care $t_1[X]=t_2[X]$, dar $t_1[YZ] \neq t_2[YZ]$. Atunci, sau $t_1[Y] \neq t_2[Y]$, sau $t_1[Z] \neq t_2[Z]$, sau ambele concomitent. Dar aceasta contrazice presupunerea, că relația r satisface dependențele $X \rightarrow Y$ şi $X \rightarrow Z$.

Exemplul 3.3. Relația r(ABCD) reprezentată în fig.3.3 satisface dependențele funcționale $A \rightarrow B$ şi $A \rightarrow C$. Conform regulii DF3, relația r trebuie să satisfacă şi dependența $A \rightarrow BC$.

DF4. Regula proiectivității. Dacă $X \to YZ$, atunci $X \to Y$.

Demonstrație. Afirmația este adevărată, fiindcă, dacă r satisface $X \to YZ$, atunci pentru orice două tupluri t_1 şi t_2 din $t_1[X]=t_2[X]$ urmează $t_1[YZ]=t_2[YZ]$ şi tuplurile vor coincide şi pe orice submulțime ale mulțimii YZ. Deci $t_1[Y]=t_2[Y]$.

Exemplul 3.4. Relația din fig.3.3 satisface dependența funcțională $A \to BC$. Conform regulii DF4, ea satisface şi dependențele $A \to B$ şi $A \to C$.

DF5. Regula tranzitivității. Dacă $X \to Y$ şi $Y \to Z$, atunci $X \to Z$.

Demonstrație. Pentru a demonstra regula DF5, vom presupune contrariul. Fie relația r satisface dependențele $X \to Y$ şi $Y \to Z$, dar nu satisface dependența $X \to Z$. Atunci relația r are cel puțin două tupluri t_1 şi t_2, pentru care $t_1[X]=t_2[X]$, iar $t_1[Z]\neq t_2[Z]$. Dacă $t_1[Y]=t_2[Y]$, atunci inegalitatea $t_1[Z]\neq t_2[Z]$ contrazice presupunerea că în r e validă dependența funcțională $Y \to Z$. Dacă $t_1[Y]\neq t_2[Y]$, atunci egalitatea $t_1[X]=t_2[X]$ contrazice presupunerea că $X \to Y$ e validă în r.

r	A	B	C	D
	a_1	b_1	c_2	d_1
	a_2	b_2	c_1	d_2
	a_3	b_1	c_2	d_1
	a_4	b_1	c_2	d_3

Fig.3.4.

Exemplul 3.5. Relația r(ABCD), reprezentată în fig.3.4, satisface dependențele funcționale $A \to B$ şi $B \to C$. Conform regulii tranzitivității, relația r satisface şi dependența funcțională $A \to C$.

DF6. Regula pseudotranzitivității. Dacă $X \to Y$ şi $YW \to Z$, atunci $XW \to Z$.

Demonstrație. Presupunem contrariul: relația r(R) satisface dependențele funcționale $X \to Y$ şi $YW \to Z$, dar nu satisface dependența funcțională $XW \to Z$. Adică există două tupluri t_1, $t_2 \in r$ pentru care $t_1[XW]=t_2[XW]$ şi $t_1[Z]\neq t_2[Z]$. Din egalitatea $t_1[XW]=t_2[XW]$ urmează $t_1[X]=t_2[X]$ şi $t_1[W]=t_2[W]$. Pot avea loc două cazuri: sau $t_1[YW]=t_2[YW]$, sau $t_1[YW]\neq t_2[YW]$.

(1) Fie $t_1[YW]=t_2[YW]$. Atunci din condiția că $t_1[Z]\neq t_2[Z]$ reiese că dependența funcțională $YW \to Z$ nu e validă în r.

(2) Fie $t_1[YW]\neq t_2[YW]$. Întrucât $t_1[W]=t_2[W]$ rezultă că $t_1[Y]\neq t_2[Y]$. Din ultima inegalitate şi din condiția $t_1[X]=t_2[X]$ urmează că relația r nu satisface dependența funcțională $X \to Y$.

Şi într-un caz şi în altul am ajuns la contradicție. Deci din $X \to Y$ şi $YW \to Z$ urmează $XW \to Z$.

Aşadar, dacă $W,X,Y,Z \subseteq R$, atunci în orice relație r cu schema R sunt valide regulile de inferență din fig.3.5.

Simbol	Denumire	Regulă

DF1	Reflexivitatea	$Y \subseteq X \Rightarrow X \rightarrow Y$
DF2	Incrementarea	$X \rightarrow Y,\ Z \subseteq W \Rightarrow XW \rightarrow YZ$
DF3	Aditivitatea	$X \rightarrow Y,\ X \rightarrow Z \Rightarrow X \rightarrow YZ$
DF4	Proiectivitatea	$X \rightarrow YZ \Rightarrow X \rightarrow Y$
DF5	Tranzitivitatea	$X \rightarrow Y,\ Y \rightarrow Z \Rightarrow X \rightarrow Z$
DF6	Pseudotranzitivitatea	$X \rightarrow Y,\ YW \rightarrow Z \Rightarrow XW \rightarrow Z$

Fig.3.5. Reguli de inferență a dependențelor funcționale

Să observăm că proiectivitatea (DF4) este, într-un sens, regula inversă regulii aditivității (DF3). Regula DF3 se utilizează pentru a uni două dependențe cu determinante egale în una, în timp ce regula DF4 - pentru descompunerea unei dependente.

3.3. Axiomele Armstrong

Definiția 3.4. Fie F o mulțime de dependențe asupra R şi fie f o dependență funcțională asupra R. *Derivație* a dependenței f din F, notată cu $F|-f$, este o consecutivitate finită de dependențe funcționale $f_1,f_2,...,f_k$ unde:

(1) orice dependență f_i poate fi dedusă din (o submulțime a mulțimii) $F \cup \{f_1,f_2,...,f_{i-1}\}$, aplicând regulile de inferență DF1-DF6;

(2) f este ultimul element, f_k, în consecutivitate.

Remarcă. Condiția (1) mai poate fi formulată în felul următor. Orice dependență f este element al mulțimii F sau se deduce din consecutivitatea $\{f_1, f_2,..., f_{i-1}\}$, aplicând regulile de inferență DF1-DF6.

Dacă $F|-f$, vom spune că F *derivă* f sau că f e *derivabilă* din F. Dacă G este mulțime de dependențe funcționale, atunci prin $F|-G$ se subînțelege că orice dependență funcțională din G e derivabilă din F.

E clar că, dacă în condiția (1) a definiției 3.4 mulțimea de dependențe funcționale F e vidă, adică $\varnothing|-f$, atunci f e dependență trivială, fiindcă singura regula de inferență corespunzătoare poate fi doar DF1.

Cu ajutorul regulilor de inferență putem deduce noi dependențe funcționale din cele date.

Exemplul 3.4. Fie r o relație cu schema R şi $X,Y,Z \subseteq R$. Presupunem că în r(R) sunt valide dependențele $XY \rightarrow Z$ şi $X \rightarrow Y$. Atunci, conform regulii DF6, relația r satisface şi dependența $XX \rightarrow Z$ care se reduce la $X \rightarrow Z$.

Pentru a combate o afirmație despre validitatea unei dependențe funcționale, e suficient de a aduce un exemplu de relație ce nu satisface afirmația dată.

r	A	B	C	D
	a_1	b_1	c_1	d_1
	a_1	b_2	c_2	d_1

Fig.3.6.

Exemplul 3.5. Să combatem afirmația că dependența $XY \to ZW$ implică dependența $X \to Z$. Relația r(ABCD) din fig.3.6 satisface dependența funcțională $AB \to CD$, dar nu satisface dependența $A \to C$.

Unele reguli de inferență pot fi deduse din altele. Armstrong a arătat că regulile DF1, DF2 şi DF5 formează o mulțime de reguli independente, iar regulile DF3, DF4 şi DF6 pot fi deduse din DF1, DF2 şi DF5. Mulțimea {DF1, DF2, DF5} de reguli de inferență este cunoscută sub denumirea de axiome Armstrong.

Teorema 3.1. Regulile DF3, DF4 şi DF6 se deduc din regulile DF1, DF2, DF5.

Demonstrație. Să arătăm că regula DF3 se deduce din regulile DF1, DF2, DF5, adică $\{X \to Y, X \to Z\} \vdash X \to YZ$, aplicând doar DF1, DF2, DF5. Într-adevăr, această aserțiune are loc, fiindcă putem construi următoarea consecutivitate de derivare.

$f_1 := X \to Y$ (dependență dată),

$f_2 := X \to XY$ ($f_1 \vdash f_2$ aplicând DF2),

$f_3 := X \to Z$ (dependență dată),

$f_4 := XY \to YZ$ ($f_3 \vdash f_4$, aplicând DF2),

$f_5 := X \to YZ$ ($\{f_2, f_4\} \vdash f_5$, aplicând DF5).

Fiindcă $X \to YZ$ este ultimul, f_5, element în consecutivitate, DF3 se deduce din DF2 şi DF5.

Să ne convingem că regula DF4 se deduce din DF1, DF2, DF5, adică $X \to YZ \vdash X \to Y$, aplicând regulile DF1, DF2, DF5. Aceasta se confirmă de următoarea consecutivitate de derivare.

$f_1 := X \to YZ$ (dependență dată),

$f_2 := YZ \to Y$ (dependență trivială, aplicând DF1),

$f_3 := X \to Y$ ($\{f_1, f_2\} \vdash f_3$, aplicând DF5).

Deci, regula DF4 se deduce din DF1 şi DF5.

În sfârşit, să arătăm că $\{X \to Y, YW \to Z\} \vdash XW \to Z$, aplicând regulile DF1, DF2 şi DF5. Putem construi următoarea consecutivitate de dependențe funcționale.

$f_1 := X \to Y$ (dependență dată),

$f_2 := XW \to YW$ ($f_1 \vdash f_2$ aplicând DF2),

$f_3 := YW \to Z$ (dependență dată),

$f_4 := XW \to Z$ ($\{f_2, f_3\} \vdash f_4$, aplicând DF5).

Deci, regula DF6 se deduce din regulile DF1, DF2, DF5.

Definiția 3.5. Fie F o mulțime de dependențe funcționale asupra schemei R şi $X, Y \subseteq R$. Închiderea mulțimii F, notată cu F^+, se defineşte recursiv:

(1) $F \subseteq F^+$;

(2) Dacă $F^1 \subseteq F$ şi $F^1 \vdash X \to Y$, atunci $X \to Y \in F^+$;

(3) Nimic altceva nu e în F^+.

Deci, $F^+ = F \cup \{X \to Y \mid F^1 \vdash X \to Y$ pentru $F^1 \subseteq F$ şi $X, Y \subseteq R\}$. Cu alte cuvinte, închiderea unei mulțimi de dependențe funcționale, F^+, reprezintă mulțimea tuturor dependențelor funcționale care se pot deriva din mulțimea F, aplicând axiomele Armstrong. Este clar că $F^+=(F^+)^+$.

Exemplul 3.6. Fie relația r(ABC) şi $F=\{AB \to C, C \to B\}$. Atunci $F^+= \{A \to A, AB \to A, AC \to A, ABC \to A, B \to B, AB \to B, BC \to B, ABC \to B, C \to C, AC \to C, BC \to C, ABC \to C, AB \to AB, ABC \to AB, AC \to AC, ABC \to AC, BC \to BC, ABC \to BC, ABC \to ABC, AB \to C, AB \to AC, AB \to BC, AB \to ABC, C \to B, C \to BC, AC \to B, AC \to AB\}$.

În F^+ primele nouăsprezece dependențe sunt triviale şi se derivă din mulțimea $\varnothing$ de dependențe, aplicând DF1, adică $\varnothing \vdash \{X \to Y \mid Y \subseteq X \subseteq ABC\}$. Alte dependențe $X \to Y$ se derivă din $Z \to Y$, unde $Z \subset X$, aplicând regula incrementării (DF2), adică $\{Z \to Y\} \vdash \{X \to Y \mid Z \subset X \subseteq R$ şi $\not\subset X$ şi $Y \subseteq R\}$. În F^+ avem şase dependențe deduse, aplicând regula DF2 asupra F. Deci $AB \to C \vdash \{AB \to AC, AB \to BC, AB \to ABC\}$ şi $\{C \to B\} \vdash \{C \to BC, AC \to B, AC \to AB\}$. Regula tranzitivității nu generează dependențe netriviale. În afară de aceasta, F^+ conține cele două dependențe din F. În total F^+ constă din douăzeci şi şapte dependențe.

Din exemplul de mai sus, se observă că numărul de dependențe ce alcătuiesc F^+ este destul de mare în raport cu F.

Dacă $F=\varnothing$, atunci F^+ constă numai din dependențe triviale. Fiindcă orice relație r(R) satisface orice dependență trivială asupra R, dependențele triviale, bineînțeles, nu se consideră constrângeri asupra relației. Întrucât R are $2^{|R|}$ submulțimi, numărul de dependențe triviale într-o relație este exponențial. Nu vom considera de asemenea dependențele de forma $X \to \varnothing$, fiindcă ele nu au aplicare practică.

3.4. Completitudinea regulilor de inferență

Definiția 3.6. Fie RI o mulțime de reguli de inferență asupra mulțimii de dependențe F. Mulțimea RI de reguli este *închisă*, dacă $F \vdash f$, utilizând regulile din RI, implică $F \models f$. Mulțimea de reguli de inferență RI este *completă*, dacă $F \models f$ implică $F \vdash f$, utilizând regulile din RI.

Că mulțimea de reguli DF1-DF6 este închisă, adică ele au loc în orice relație, s-a demonstrat pentru fiecare regulă în parte în secțiunea 3.2. Pentru a arăta că mulțimea de reguli este completă mai întâi introducem noțiunea de închidere a unei mulțimi de atribute.

Definiția 3.7. Fie F o mulțime de dependențe asupra R şi $X \subseteq R$. Închiderea mulțimii de atribute X în raport cu mulțimea de dependențe F, notată cu X^+, se defineşte astfel:

(1) $X \subseteq X^+$ (X e o submulțime a închiderii);
(2) Dacă $Z \subseteq X^+$ şi $Z \to Y \in F$, atunci $Y \subseteq X^+$;
(3) Nici un alt atribut nu face parte din X^+.

Adică $X^+=X\cup\{Y|Z\subseteq X^+$ şi $Z\rightarrow Y\in F\}$.

Condiție definiția 3.6.	Comparare atribute, dependențe	$(AB)^+$
(1)	$AB\subseteq(AB)^+$	AB
(2)	$AB\subseteq AB$ şi $AB\rightarrow DE\in F$	ABDE
(2)	$D\subseteq ABDE$ şi $D\rightarrow C\in F$	ABCDE
(2)	$CE\subseteq ABCDE$ şi $CE\rightarrow G\in F$	ABCDEG

Fig.3.7.

Exemplul 3.7. Fie R=ABCDEG şi $F=\{D\rightarrow C, AB\rightarrow DE, CE\rightarrow G\}$. Să se arate că $(AB)^+$ = ABCDEG. În fig.3.7 este reprezentat procesul de construire a $(AB)^+$.

r	X^+	$R\backslash X^+$
t_1	a a ... a	a a ... a
t_2	a a ... a	b b ... b

Fig.3.8.

Teorema 3.2. Mulțimea de reguli de inferență DF1-DF6 este completă.

Demonstrație. Fie F o mulțime de dependențe asupra R. Trebuie să arătăm că, dacă $F|=X\rightarrow Y$, atunci $F|-X\rightarrow Y$ (sau, ce e echivalent, dacă $F|-X\rightarrow Y$, atunci $F|\neq X\rightarrow Y$).

Fie $X\rightarrow Y$ o dependență funcțională ce nu se deduce din F, adică $F|-X\rightarrow Y$. Să arătăm că relația ce satisface toate dependențele din F nu satisface $X\rightarrow Y$, adică $F|\neq X\rightarrow Y$.

Definim relația r(R) în felul următor (vezi fig.3.8). Ea constă din două tupluri t_1 şi t_2. Tuplul t_1=aa...a; tuplul t_2 se defineşte astfel $t_2[A]$ = {a, dacă $A_i\in X^+$; b, dacă $A_i\in R\backslash X^+$}.

Mai întâi, vom arăta că relația construită satisface toate dependențele din F. Presupunem contrariul. Există în F o dependență $V\rightarrow W$ ce nu e validă în r(R). Atunci $V\subseteq X^+$ şi $W\not\subset X^+$, altminteri relația r va satisface dependența $V\rightarrow W$. Întrucât W□X^+, există în W cel puțin un atribut A şi $A\notin X^+$. Conform regulii reflexivității $V\subseteq X^+$ implică $X^+\rightarrow V$. Dependențele $X^+\rightarrow V$ şi $V\rightarrow W$ implică, conform regulii DF5, $X^+\rightarrow W$. Din $A\in W$ urmează $W\rightarrow A$. Aplicând asupra $X^+\rightarrow W$ şi $W\rightarrow A$ regula tranzitivității obținem $X^+\rightarrow A$. Din ultima dependență, $X^+\rightarrow A$, urmează că $A\in X^+$, ce contrazice faptului că $A\notin X^+$. Deci relația construită satisface orice dependență din F.

Acum să arătăm că relația noastră nu satisface dependența $X\rightarrow Y$. Presupunem că relația r(R) satisface dependența $X\rightarrow Y$. Atunci $t_1[X]=t_2[X]$ implică $t_1[Y]=t_2[Y]$. Aceasta se poate întâmpla doar când $Y\subseteq X^+$. Din $Y\subseteq X^+$, aplicând regula DF1, urmează $X^+\rightarrow Y$. Dependența $X\rightarrow X^+$ este în F^+. Aplicând asupra $X\rightarrow X^+$ şi $X^+\rightarrow Y$ regula DF5, obținem $X\rightarrow Y$. Dar aceasta contrazice ipoteza că $F|-X\rightarrow Y$.

Deci, dacă $F|=X\rightarrow Y$, atunci $F|-X\rightarrow Y$.

Luând în considerație că (în compartimentul 3.2.) s-a demonstrat că mulțimea de reguli este închisă, adică $F|-X\to Y$ implică $F|=X\to Y$ putem spune că mulțimea de reguli de inferență DF1-DF6 este închisă şi completă, adică $F|=X\to Y$, dacă şi numai dacă $F|-X\to Y$. Prin urmare, putem utiliza "|= " şi "|-" deopotrivă.

Dat fiind faptul că regulile DF1-DF6 se deduc din axiomele Armstrong, vom spune că şi axiomele Armstrong sunt închise şi complete.

3.5. Modele de derivări

Noțiunea de consecutivitate de derivare introdusă în secțiunea 3.3 are o serie de dezavantaje. De regulă, pentru o dependență f pot exista o serie de derivări din mulțimea de dependențe date F. Aceste derivări, fiind în esență echivalente, diferă prin ordinea şi tipul regulilor aplicate pentru derivarea dependențelor din consecutivitate. În plus, consecutivitățile de derivare pot conține dependențe derivate redundante. Mai jos vom examina modele de derivări ce într-o măsură sau alta sunt lipsite de aceste dezavantaje.

3.5.1. RAP–consecutivități de derivare

Axiomele Armstrong sunt o submulțime completă de reguli de inferență a mulțimii DF1-DF6. Există, însă, mulțimi complete de reguli de inferență ce nu sunt submulțimi ale mulțimii DF1-DF6. Considerăm o astfel de mulțime de reguli de inferență denumită B-axiome.

Pentru relația r(R), submulțimile W, X, Y şi Z ale mulțimii R şi $C\in R$ avem:

B1. Regula reflexivității. Dacă $Y\subseteq X$, atunci $X\to Y$.

B2. Regula acumulării. Dacă $X\to YZ$ şi $Z\to CW$, atunci $X\to YZC$.

B3. Regula proiectivității. Dacă $X\to YZ$, atunci $X\to Y$.

Teorema 3.3. Mulțimea de reguli B1-B3 este o mulțime închisă.

Demonstrație. În secțiunea 3.2 s-a demonstrat că regulile reflexivității şi proiectivității au loc în orice relații r cu schema R. Să examinăm regula acumulării. Presupunem contrariul: relația r satisface dependențele $X\to YZ$, $Z\to CW$ şi nu satisface $X\to YZC$. Atunci există două tupluri t_1 şi t_2 pentru care $t_1[X]=t_2[X]$, dar $t_1[YZC]\neq t_2[YZC]$. Din $t_1[YZC]\neq t_2[YZC]$ urmează sau $t_1[YZ]\neq t_2[YZ]$, sau $t_1[C]\neq t_2[C]$. Dacă $t_1[YZ]\neq t_2[YZ]$, atunci dependența $X\to YZ$ nu e validă în r. Dacă $t_1[C]\neq t_2[C]$ atunci r nu satisface dependența $Z\to CW$. Prin urmare, dependența $X\to YZC$ e validă în orice relație, în care sunt valide dependențele $X\to YZ$ şi $Z\to CW$.

Teorema 3.4. Mulțimea de reguli B1-B3 este completă.

Demonstrație. Pentru a arăta că regulile B1-B3 sunt complete, e de ajuns să arătăm că axiomele Armstrong se deduc din B-axiome.

Regula reflexivității DF1 coincide cu B1.

Să arătăm că regula incrementării, DF2, urmează din regulile B1-B3. Fie $X\to Y$. Din regula B1 urmează $XZ\to XZ$. Aplicând asupra $XZ\to XZ$ şi $X\to Y$ regula B2 de atâtea ori câte atribute sunt în Y, obținem $XZ\to XZY$. Conform regulii proiectivității, B3, obținem $XZ\to Y$.

Să demonstrăm că regula tranzitivității DF5 urmează din B1-B3. Fie relația r satisface dependențele funcționale $X \rightarrow Y$ şi $Y \rightarrow Z$. Din B1 obținem $X \rightarrow X$. Consecutiv, aplicăm mai întâi regula B2 asupra $X \rightarrow X$ şi $X \rightarrow Y$ de atâtea ori câte atribute sunt în Y şi obținem $X \rightarrow XY$. Aplicând asupra $X \rightarrow XY$ şi $Y \rightarrow Z$, regula B2, obținem $X \rightarrow XYZ$. O singură aplicare a regulii B3 produce $X \rightarrow Y$.

Definiția 3.8. Consecutivitatea de dependențe funcționale se numeşte RAP-consecutivitate de derivare (după primele litere ale denumirilor B-axiomelor: **R**eflexivitate, **A**cumulare, **P**roiectivitate) a unei dependențe $X \rightarrow Y$ din mulțimea de dependențe F, dacă:

(1) prima dependență în consecutivitate e $X \rightarrow X$;

(2) ultima dependență în consecutivitate e $X \rightarrow Y$ (obținută, aplicând regula B3);

(3) orice dependență din consecutivitate (în afară de prima şi ultima) sau este o dependență din F, sau are forma $X \rightarrow Z$ şi e obținută, aplicând regula B2 asupra dependențelor precedente.

Exemplul 3.8. Fie $F=\{AB \rightarrow E, AG \rightarrow J, BE \rightarrow I, E \rightarrow G, GI \rightarrow H\}$. Consecutivitatea de mai jos este RAP-consecutivitate de derivare a dependenței $AB \rightarrow GH$ din F.

$f_1:=AB \rightarrow AB$	(B1),
$f_2:=AB \rightarrow E$	(dată),
$f_3:=AB \rightarrow ABE$	(B2: f_1, f_2),
$f_4:=BE \rightarrow I$	(dată),
$f_5:=AB \rightarrow ABEI$	(B2: f_3, f_4),
$f_6:=E \rightarrow G$	(dată),
$f_7:= AB \rightarrow ABEIG$	(B2:f_5, f_6),
$f_8:=GI \rightarrow H$	(dată),
$f_9:=AB \rightarrow ABEIGH$	(B2: f_7, f_8),
$f_{10}:=AB \rightarrow GH$	(B3: f_9).

3.5.2. DDA-grafuri de derivare

DDA-graful (Derivation Directed Acyclic-graph) este o interpretare grafică a RAP-consecutivității de derivare.

Definiția 3.9. Fie F o mulțime de dependențe funcționale asupra schemei R. DDA-graf asupra F este un graf orientat fără cicluri ce se defineşte recursiv:

R1. O mulțime de noduri notate cu atribute din R este DDA-graf.

R2. Dacă H este DDA-graf şi $B_1,...,B_n$ sunt noduri în H şi dependența funcțională $B_1...B_n \rightarrow CZ$ este în F, atunci H^1, obținut din H prin adăugarea nodului C (dacă astfel de nod nu există) şi muchiilor (B_1C) ... (B_nC) orientate spre C, este DDA-graf.

R3. DDA-graf este numai graful obținut prin aplicarea regulilor R1 şi R2.

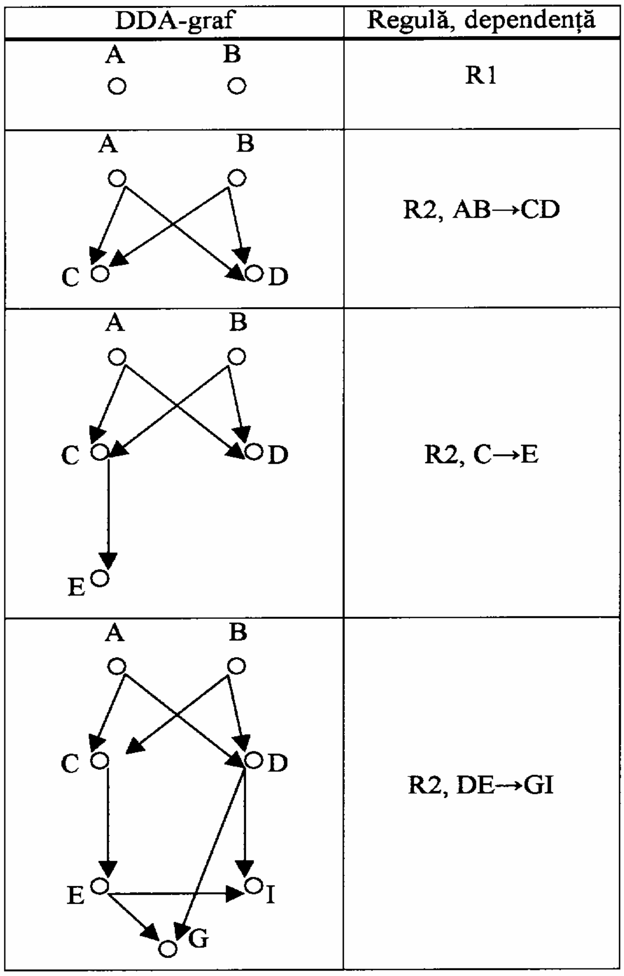

Fig.3.9.

Definiția 3.10. Nodul B al unui DDA-graf H se numeşte *inițial,* dacă în el nu intră nici o muchie (Nodurile inițiale se adaugă în H prin regula R1).

Definiția 3.11. Fie H un DDA-graf asupra F. Graful H se numeşte DDA-graf de derivare a dependenței X→Y din F, dacă:

(1) X este mulțimea de noduri inițiale în H;

(2) orice atribut din Y este un nod în H.

Definiția 3.12. Mulțimea de dependențe din F utilizate de regula R2 pentru construirea DDA-grafului H de derivare a unei dependențe X→Y din F se numeşte *mulțime utilizabilă*, notată cu U(H).

Exemplul 3.9. În fig.3.9 sunt prezentate etapele de construire a DDA-grafului de derivare a dependenței AB→CG din mulțimea de dependențe funcționale F={AB→CD, A→I, C→E, DE→GI}.

Nodurile inițiale sunt A şi B. Mulțimea utilizabilă U(H)={$AB\rightarrow CD$, $C\rightarrow E$, $DE\rightarrow GI$}. Graful H obținut este DDA-graful de derivare a dependenței $AB\rightarrow CG$ din F, fiindcă mulțimea de atribute AB formează nodurile inițiale din H, iar CG sunt noduri în H.

3.5.3. Derivația maximală

Modelele de derivare descrise mai sus poartă mai mult un caracter teoretic. Ele nu sunt lipsite de neajunsul că pentru o dependență dată există mai multe consecutivități de derivare. Aici vom considera un model, numit derivare maximală, liber de acest dezavantaj. Acest model este foarte aproape de noțiunea de închidere a unei mulțimi de atribute în raport cu o mulțime de dependențe funcționale. El va fi utilizat în demonstrarea diverselor rezultate privind acoperirile de dependențe funcționale.

Definiția 3.13. Fie F o mulțime de dependențe funcționale asupra schemei R şi fie $X\subseteq R$. Derivația maximală a mulțimii de atribute X în raport cu mulțimea de dependențe funcționale F este o consecutivitate de mulțimi de atribute $<X_0,X_1,...,X_n>$, unde

(1) $X_0=X$;

(2) $X_i=X_{i-1}\cup Z$, $1 \le i \le n$, unde $Z = \cup_j W_j$ pentru orice dependență $V_j\rightarrow W_j\in F$ ce satisface $V_j\subseteq X_{i-1}$ şi $W_j\not\subset X_{i-1}$;

(3) în F nu există nici o dependență $V_j\rightarrow W_j$ pentru care $V_j\subseteq X_n$ şi $W\not\subset X_n$.

Înainte de a arăta că derivația maximală este un instrument puternic de modelare a derivării dependențelor funcționale, considerăm două proprietăți ale ei.

Lema 3.1. Dacă $X\subseteq Y$ şi consecutivitățile $<X_0, X_1,..., X_n>$, $<Y_0, Y_1,..., Y_m>$ sunt derivații maximale ale mulțimilor X şi Y, corespunzător, în raport cu F, atunci pentru orice X_i există o mulțime Y_j încât $X_i\subseteq Y_j$ şi $j\le i$.

Demonstrație. Vom arăta aceasta, aplicând inducția matematică asupra i. Când i=0 avem $X_0\subseteq Y_0$, fiindcă $X\subseteq Y$. Fie că presupunerea noastră e justă pentru i=k: $X_k\subseteq Y_p$ şi $p\le k$. Să arătăm că ceea ce trebuie de demonstrat are loc şi pentru i=k+1. Într-adevăr, la pasul k+1, $X_{k+1}=X_k\cup Z$, unde $Z=\cup_j W_j$ pentru toate dependențele $V_j\rightarrow W_j$ din F determinanții şi determinații cărora satisfac $V_j\subseteq X_k$ şi $W_j\not\subset X_k$ corespunzător. Conform ipotezei inducției $X_k\subseteq Y_p$. Prin urmare, toți determinanții dependențelor $V_j\rightarrow W_j$ ce se conțin în X_k se vor conține şi în Y_p . Dat fiind faptul că mulțimea Y_p e mai "largă" ea poate conține toți determinații W_j şi atunci $X_{k+1}\subseteq Y_p$. Dacă nu, atunci în derivația mulțimii Y în raport cu F se execută următorul p+1 pas, în rezultatul căruia vom obține Y_{p+1} care va conține X_{k+1}.

Lema 3.2. Dacă $<X_0, X_1,..., X_n>$ este derivația maximală a mulțimii X în raport cu mulțimea de dependențe funcționale F, atunci $X\rightarrow X_i\in F^+$, $0\le i\le n$.

Demonstrație. Vom face demonstrarea, utilizând inducția asupra numărului de aplicări a regulii (2) în construirea derivației maximale.

Fie că în construirea derivației maximale nu s-a aplicat regula (2). Atunci ea are un singur element X_0, unde $X_0=X$ şi conform regulii reflexivității, DF1, $X\rightarrow X_0\in F^+$.

Presupunem că la aplicarea i-1 a regulii (2) are loc $X\rightarrow X_{i-1}\in F^+$. Să se arate veridicitatea afirmației pentru pasul i. Fără a constrânge generalitatea, presupunem că la acest pas avem o singură dependență $V\rightarrow W$ ce satisface $V\subseteq X_{i-1}$, $W\not\subset X_{i-1}$. Conform regulii reflexivității $X_{i-1}\rightarrow V\in F^+$. Dar $X_{i-1}\rightarrow V\in F^+$ şi $V\rightarrow W\in F^+$ (regula tranzitivității) implică $X_{i-1} \rightarrow W \in F^+$. Adăugăm la determinantul și determinatul ultimei dependențe mulțimea X_{i-1}. Obținem (conform regulii incrementării) $X_{i-1} \rightarrow X_{i-1}W \in F^+$. Din $X \rightarrow X_{i-1} \in F^+$ (ipoteza inducției) și $X_{i-1} \rightarrow X_i \in F^+$ urmează $X \rightarrow X_i \in F^+$.

În baza acestor două proprietăți vom demonstra

Teorema 3.5. Fie $<X_0, X_1, ..., X_n>$ derivația maximală a mulțimii X în raport cu mulțimea de dependențe funcționale F. Atunci $X \rightarrow Y \in F^+$ dacă și numai dacă $Y \subseteq X_n$.

Demonstrație. *Necesitatea.* Să arătăm că, dacă $X \rightarrow Y \in F^+$, atunci există o mulțime X_i în derivația maximală $<X_0, X_1, ..., X_n>$ încât $Y \subseteq X_i$ și, prin urmare, $Y \subseteq X_n$. Vom utiliza inducția asupra (lungimii derivării) numărului de dependențe folosite în derivarea dependenței $X \rightarrow Y$ în raport cu F, unde dependența de derivare sau este în F, sau se deduce din regula reflexivității sau din regula incrementării, aplicate asupra unei dependențe precedente, sau cu ajutorul regulii tranzitivității, aplicate asupra a două dependențe precedente. Ultima dependență în derivare, bineînțeles, e $X \rightarrow Y$.

Fie că derivarea dependenței $X \rightarrow Y$ are lungimea 1, adică constă din însuși $X \rightarrow Y$. Sunt două cazuri. Sau $X \rightarrow Y$ se deduce din axioma reflexivității, sau $X \rightarrow Y \in F$. În primul caz, $Y \subseteq X$ și, prin urmare, $Y \subseteq X_0$. În al doilea caz, dependența $X \rightarrow Y$ va participa la formarea elementului al doilea a derivației maximale a mulțimii X în raport cu F. Deci $Y \subseteq X_1$.

Presupunem acum că afirmația noastră e justă pentru o derivare cu lungimea mai mică decât k și să demonstrăm veridicitatea afirmației noastre pentru derivarea cu lungimea k. Considerăm consecutiv regulile de inferență ce pot fi aplicate la acest pas.

Dacă pentru deducerea dependenței $X \rightarrow Y$ se aplică regula reflexivității sau $X \rightarrow Y \in F$, atunci Y se comportă ca și pentru derivări cu lungimea unu, adică Y se pomenește corespunzător în X_0 și X_1.

Dacă $X \rightarrow Y$ urmează din regula incrementării asupra unei dependențe precedente $V \rightarrow W$, atunci există S și T, unde $T \subseteq S$ și $VS=X$, $WT=Y$. Întrucât $V \rightarrow W$ are o derivare cu o lungime mai mică decât k, atunci conform ipotezei inducției există în derivația maximală o mulțime V_j, unde $W \subseteq V_j$. Fiindcă $V \subseteq X$, atunci conform lemei 3.1 există în derivația maximală pentru X în raport cu F o mulțime X_i, unde $W \subseteq X_i$. Din $T \subseteq S \subseteq X$ urmează $T \subseteq X_0$ și $T \subseteq X_i$.

Considerăm ultimul caz, când dependența $X \rightarrow Y$ e obținută, aplicând regula tranzitivității asupra a două dependențe precedente $X \rightarrow Z$ și $Z \rightarrow Y$, derivațiile cărora au lungimi mai mici decât k.

Urmând ipoteza inducției, pentru $X \rightarrow Z$ și $Z \rightarrow Y$ avem corespunzător $Z \subseteq X_j$ și $Y \subseteq Z_p$. Însă Z_p este element al derivației maximale a mulțimii Z în raport cu F. Fiindcă $Z \subseteq X_j$, conform lemei 3.1 $Z_p \subseteq X_{j+m}$, unde X_{j+m} este elementul m+1 al derivației maximale a mulțimii X_j în raport cu F pe care o vom nota $<X_{j+0}, X_{j+1}, ..., X_{j+m}, ..., X_{j+p}>$. Este evident că derivația maximală a mulțimii X_j nu e altceva decât o subconsecutivitate ce constă din ultimele n-j+1 elemente ale derivației maximale a mulțimii X în raport cu F. Prin urmare, $Y \subseteq X_i$, unde i=j+m.

Suficiența. Fie $<X_0, X_1, ..., X_n>$ e derivația maximală a mulțimii X în raport cu F. Conform lemei 3.2 $X \rightarrow X_n \in F^+$. Întrucât $Y \subseteq X_n$, atunci aplicând regula proiectivității asupra dependenței $X \rightarrow X_n$ obținem $X \rightarrow Y \in F^+$. Teorema e demonstrată.

Definiția 3.14. Fie $X \rightarrow Y \in F^+$ şi $<X_0, X_1,..., X_n>$ derivația maximală a mulțimii X în raport cu F. Fie X_i este primul element din consecutivitate ce conține mulțimea Y. Subconsecutivitatea $<X_0, X_1,..., X_i>$ se numeşte *derivația* dependenței funcționale $X \rightarrow Y$ în raport cu F.

Din teorema 3.5 şi definiția 3.14 urmează

Consecința 3.1. $X \rightarrow Y \in F^+$ atunci şi numai atunci când există derivația dependenței $X \rightarrow Y$ în raport cu F.

Consecința 3.2. Dacă $X \rightarrow Y \in F^+$ şi dependența $V \rightarrow W \in F$ e utilizată în construirea derivației dependenței $X \rightarrow Y$ în raport cu F, atunci $X \rightarrow V \in F^+$.

Justețea acestei afirmații decurge imediat din lema 3.2 şi regula proiectivității.

Trebuie menționat că derivația maximală este un model de derivare liber de dezavantajele menționate la începutul acestei secțiuni. Existența a unei singure derivații pentru o dependență dată va fi utilă în expunerea de mai departe a materiei.

3.5.4. Algoritmi

Pentru a determina dacă $F|=X \rightarrow Y$, e suficient de verificat dacă $X \rightarrow Y \in F^+$. Însă, F^+ este excesiv de mare în raport cu F. E dezirabilă o metodă de verificare, dacă $X \rightarrow Y$ aparține F^+, fără a deduce toate dependențele funcționale din F. Un astfel de algoritm e prezentat mai jos. Nucleul algoritmului constă din procedura de construire a închiderii mulțimii de atribute X în raport cu F. După ce se găseşte X^+, se verifică dacă $Y \subseteq X^+$.

Este evident că ultimul element, X_n, din derivația maximală nu este altceva decât X^+. Iar teorema 3.5 ne sugerează că $X \rightarrow Y$ urmează logic din F, dacă $Y \subseteq X^+$. Deci, derivația maximală serveşte drept model teoretic pentru următorul algoritm de determinare a lui X^+.

Algoritmul CLOSURE caută în F o dependență funcțională pentru care determinantul reprezintă o submulțime a lui X_i, iar determinatul nu este inclus în X_i. Dacă se găseşte o astfel de dependență funcțională, atunci se adaugă la X_i atributele care constituie determinatul dependenței. Dacă nu se găseşte, atunci închiderea căutată, X^+ este reprezentată de mulțimea de atribute X_i.

Algoritmul CLOSURE (F, X, X^+)

Intrare: F- o mulțime de dependențe funcționale asupra schemei R; X – o mulțime de atribute, $X \subseteq R$.

Ieşire: X^+ - închiderea mulțimii X în raport cu F

```
begin
      i:=0; Xi:=X;
      repeat
            i:=i+1;
            Xi:=Xi-1;
            For all V→W in F
               if V⊆Xi  then Xi:=Xi∪W;
      until Xi=Xi-1;
      return (X+:=Xi);
end
```

Exemplul 3.10. Fie $F=\{B\rightarrow CD, AD\rightarrow E, B\rightarrow A\}$, $X=B$. Să se calculeze X^+.
Inițial $X_0=B$.
În ciclul repeat:
$X_1=B$.
În ciclul for:
$X_1=BCD$ (aplicând $B\rightarrow CD$),
$X_1=ABCD$ (aplicând $B\rightarrow A$).
$X_2=ABCD$
În ciclul for:
$X_2=ABCDE$ (aplicând $AD\rightarrow E$)
$X_3=ABCDE$
După ciclul for:$X_3=X_2$.
Rezultat: $X^+:=ABCDE$.
Deci închiderea lui B în raport cu F este $(B)^+=ABCDE$.

Algoritmul CLOSURE realmente construiește derivația maximală a mulțimii de atribute X în raport cu F. Apelând la CLOSURE e ușor de construit algoritmul de verificare a apartenenței unei dependențe funcționale la F^+. Această verificare e realizată în algoritmul MEMBERSHIP.

Algoritmul MEMBERSHIP(F, X→Y)
Intrare: F- o mulțime de dependențe funcționale;
$X\rightarrow Y$ – o dependență funcțională.
Ieșire: adevăr, dacă $F|=X\rightarrow Y$, fals - în caz contrar.
begin
CLOSURE (F, X, X^+);
if $Y\subseteq X^+$ then return (adevăr) else return (fals);
end

Corectitudinea algoritmilor CLOSURE și MEMBERSHIP rezultă imediat din teorema 3.5.

3.6. Acoperiri

În această secțiune se consideră diverse moduri de reprezentare a mulțimilor de dependențe, cum ar fi mulțimile nonredundante, reduse, canonice, minimale și optimale.

3.6.1. Mulțimi echivalente de dependențe funcționale

Definiția 3.15. Două mulțimi de dependențe funcționale F și G se numesc *echivalente*, notat cu $F\equiv G$, dacă $F^+=G^+$. Vom mai spune în acest caz că F *acoperă* G (sau G acoperă F).

Dacă $F\equiv G$, adică $F^+=G^+$, atunci orice dependență $X\rightarrow Y$ ce urmează logic din F urmează și din G. Deci pentru a verifica dacă F și G sunt echivalente se ia orice dependență $X\rightarrow Y$ din F și se verifică dacă $G|=X\rightarrow Y$. Dacă o oarecare dependență $X\rightarrow Y$ nu aparține lui G^+, atunci $F^+\neq G^+$. Apoi analogic se verifică dacă orice dependență

$V \to W$ din G se deduce din F. Dacă toate dependențele se deduc, mulțimile F şi G sunt echivalente.

Exemplul 3.11. Mulțimile $F=\{AB \to C, AC \to D, AD \to B, C \to B\}$ şi $G=\{AD \to C, AB \to D, C \to B\}$ sunt echivalente, însă F nu este echivalentă mulțimii $G^1=\{AB \to C, AC \to D, AD \to B, AC \to B\}$ (a se verifica în calitate de exercițiu).

3.6.2. Acoperiri nonredundante

Definiția 3.16. Mulțimea de dependențe funcționale F este *nonredundantă,* dacă nu există o submulțime proprie F^1 a mulțimii F şi $F^1 \equiv F$. Dacă o astfel de submulțime există, atunci F se numeşte *redundantă*. Mulțimea F este *acoperire nonredundantă* a mulțimii G, dacă F este acoperire pentru G şi F este nonredundantă.

Exemplul 3.12. Fie $G=\{A \to BC, B \to C\}$. Mulțimea $F=\{A \to B, A \to C, B \to C\}$ este acoperire a mulțimii G, dar nu e acoperire nonredundantă, fiindcă $F^1=\{A \to B, B \to C\}$ e acoperire pentru G, însă $F^1 \subset F$.

Să considerăm o altă interpretare a noțiunii de mulțime nonredundantă.

Definiția 3.17. Mulțimea F de dependențe funcționale se numeşte nonredundantă, dacă în ea nu există nici o dependență $X \to Y$ încât $(F \setminus \{X \to Y\}) \models X \to Y$. În caz contrar, F se numeşte redundantă.

Această definiție este pusă în baza următorului algoritm de construire a acoperirii nonredundante. E de menționat că rezultatul obținut în urma aplicării algoritmului depinde de ordinea considerării dependențelor funcționale.

Algoritmul NONREDUN (F,G)

```
Intrare: F – o mulțime de dependențe funcționale.
Ieşire: G – o acoperire nonredundantă a mulțimii F.
begin
    G:=F;
    for all X→Y in G
        if MEMBERSHIP (G \ {X→Y}, X→Y) then
        G:=G \ {X→Y};
    return (G);
end
```

Exemplul 3.13. Fie $F = \{A \to B, B \to A, B \to C, A \to C\}$. În rezultatul aplicării algoritmului NONREDUN obținem acoperirea nonredundantă $G = \{A \to B, B \to A, A \to C\}$. Dacă mulțimea F e prezentată în altă ordine $\{A \to B, A \to C, B \to A, B \to C\}$ se obține rezultatul $G = \{A \to B, B \to A, B \to C\}$.

3.6.3. Acoperiri reduse

Dacă F e o mulțime nonredundantă, atunci nu poate fi eliminată din F nici o dependență funcțională fără a afecta echivalența mulțimii obținute cu cea anterioară. În schimb poate fi micşorată dimensiunea mulțimii F, eliminând unele atribute din dependențele funcționale.

Definiția 3.18. Fie F o mulțime de dependențe funcționale asupra schemei R şi $X \rightarrow Y \in F$. Atributul A este *redundant* în dependența $X \rightarrow Y$ în raport cu F, dacă

(1) $A \in X$, $V = X \setminus A$ şi $F \setminus \{X \rightarrow Y\} \cup \{V \rightarrow Y\} \equiv F$

sau

(2) $A \in Y$, $W = Y \setminus A$ şi $F \setminus \{X \rightarrow Y\} \cup \{X \rightarrow W\} \equiv F$.

Cu alte cuvinte, atributul A este redundant în dependența $X \rightarrow Y$, dacă el poate fi eliminat din determinant sau determinat, fără a fi schimbată închiderea mulțimii F. Procesul de eliminare a atributelor redundante se numeşte, corespunzător, reducere în stânga şi reducere în dreapta a dependențelor.

Exemplul 3.14. Fie $F = \{AC \rightarrow B, A \rightarrow C, A \rightarrow BD\}$. Atributul C este redundant în $AC \rightarrow B$, fiindcă $(AC)^+ = A^+ = ACBD$. Adică $A \rightarrow B$ este în F^+. Deci dependența $AC \rightarrow B$ poate fi înlocuită cu $A \rightarrow B$ în mulțimea de dependențe funcționale F. Atributul B este redundant în partea dreaptă a dependenței $A \rightarrow BD$.

Definiția 3.19. Fie F o mulțime de dependențe funcționale asupra schemei R. Mulțimea F se numeşte *redusă în stânga* (dreapta), dacă orice dependență din F nu are atribute redundante în partea stângă (dreaptă). Mulțimea de dependențe redusă în stânga şi în dreapta se numeşte *redusă*.

Exemplu 3.15. Mulțimea $F = \{AC \rightarrow B, A \rightarrow C, A \rightarrow BD\}$ nu este redusă nici în stânga, nici în dreapta. Mulțimea $F_1 = \{A \rightarrow B, A \rightarrow C, A \rightarrow BD\}$ e redusă în stânga şi nu e redusă în dreapta, dar $F_2 = \{AC \rightarrow B, A \rightarrow C, A \rightarrow D\}$ e redusă în dreapta şi nu în stânga. Mulțimea de dependențe funcționale $F_3 = \{A \rightarrow B, A \rightarrow C, A \rightarrow D\}$ e redusă în stânga şi în dreapta, deci e redusă.

Mai jos se aduc algoritmii de reducere a unei mulțimi nonredundante.

Algoritmul LEFTRED (F, G)

Intrare: F – o mulțime nonredundantă de dependențe funcționale.
Ieşire: G – o mulțime de dependențe funcționale redusă în stânga.

```
begin
    G:=F;
    for all X→Y in G
        for all A in X
            if MEMBERSHIP (G, (X \ A) →Y) then
            G:=G \ {X→Y}∪{(X \ A)→Y};
    return (G);
end
```

Algoritmul RIGHTRED (F,G)

Intrare: F – o mulțime nonredundantă de dependențe funcționale.
Ieşire: G – o mulțime de dependențe funcționale redusă în dreapta

```
begin
    G:=F;
    for all X→Y in G
```

```
        for all A in Y
            if MEMBESHIP (G \ {X→Y}∪{X→(Y \ A)}, X→A) then G:= {G \
            {X→Y}∪{X→(Y \ A)};
    return (G);
end
```

Algoritmul REDUCE (F, G)
Intrare: F – o mulțime de dependențe funcționale nonredundantă.
Ieşire: G – o mulțime redusă de dependențe de dependențe funcționale.

```
begin
    LEFTRED (F, F¹);
    RIGHTRED (F¹, G);
    eliminarea din G a dependențelor X→∅;
    return (G);
end
```

În algoritmul LEFTRED de mai sus, este inclusă condiția de verificare $G|= (X \setminus A) \rightarrow Y$. Este evident că, dacă $(X\setminus A)\rightarrow Y$ se deduce din G, atunci $X\rightarrow Y$ poate fi substituită cu $(X\setminus A)\rightarrow Y$, fiindcă $(X \setminus A)\rightarrow Y|=X\rightarrow Y$.

E evident că, dacă o dependență este redundantă, atunci toate atributele ei sunt redundante. Pentru a evita apariția dependențelor de forma $\varnothing\rightarrow\varnothing$ se presupune că mulțimea destinată reducerii este nonredundantă.

S-ar părea că acoperirile reduse pot fi calculate, găsind şi eliminând în mod aleator atributele redundante. Însă, examinând părțile stângi şi drepte ale dependențelor în ordine diferită, obținem rezultate diferite. În părțile drepte odată examinate pot apărea atribute redundante după examinarea părților stângi. Deci eliminarea atributelor redundante trebuie să înceapă cu partea stângă. Dar şi în cazul acesta pot apărea dependențe de forma $X\rightarrow\varnothing$. Ele se elimină la urmă din mulțimea rezultat.

3.6.4. Acoperiri canonice

Definiția 3.20. Mulțimea de dependențe funcționale F este *canonică,* dacă F este nonredundantă, redusă în stânga şi orice dependență din F are forma $X\rightarrow A$.

Întrucât mulțimea canonică este nonredundantă şi redusă în stânga, iar determinatul oricărei dependente constă dintr-un singur atribut, ea este redusă şi în dreapta, adică este redusă.

Exemplul 3.16. Mulțimea $F=\{A\rightarrow B, A\rightarrow C, B\rightarrow D\}$ este o acoperire canonică a mulțimii $G=\{A\rightarrow BC, B\rightarrow D\}$.

Teorema 3.6. Fie F o acoperire redusă. Se formează mulțimea G, dezagregând orice dependență de forma $X\rightarrow A_1...A_n$ în $X\rightarrow A_1,..., X\rightarrow A_n$. Atunci G este canonică. Fie G o acoperire canonică. Formăm F, agregând dependențele cu determinanți egali. Mulțimea F este acoperire redusă.

Demonstrație. Fie mulțimea G este formată din F prin dezagregarea dependențelor. Presupunem că G nu e canonică. Dacă dependența $X\rightarrow A_i$ e redundantă, atunci atributul A_i e redundant în $X \rightarrow A_1...A_n$. Dacă $X \rightarrow A_i$ conține un atribut redundant în X, fie B, atunci $G|=(X \backslash B) \rightarrow A_i$. Dar $G|=(X \backslash B) \rightarrow X$, fiindcă $X \rightarrow A_i$ e nonredundantă. De unde conchidem că $F|=(X \backslash B) \rightarrow X$ şi, prin urmare, B e redundant în partea stângă a dependenței $X \rightarrow A_1...A_n$ din F.

Să demonstrăm a doua parte a teoremei. Presupunem contrariul: F nu e redusă. Dacă F nu e redusă în dreapta, atunci G nu e nonredundantă. Dar dacă F nu e redusă în stânga, atunci şi G nu e redusă din stânga, ce contravine presupunerii că G e canonică.

3.6.5. Clase de echivalență

Definiția 3.21. Fie F o mulțime de dependențe funcționale asupra schemei R şi X, $Y \subseteq R$. Mulțimile de atribute X şi Y sunt *echivalente*, notăm cu $X \leftrightarrow Y$, în raport cu F, dacă $F|=X \rightarrow Y$ şi $F|=Y \rightarrow X$.

Această definiție ne sugerează că mulțimea F poate fi partiționată în clase de echivalență. Adică asupra F se poate defini o relație de echivalență: dependențele $X \rightarrow Y$ şi $V \rightarrow W$ din F aparțin unei clase de echivalență, dacă şi numai dacă $X \leftrightarrow V$ în raport cu F.

Notăm cu $E_F(X)$ mulțimea de dependențe cu determinanții echivalenți lui X în raport cu F, adică $E_F(X)=\{V \rightarrow W|\ F|=V \rightarrow X\ \&\ F|=X \rightarrow V\}$. $E_F(X)$ se numeşte *clasă de echivalență* a dependențelor cu determinanții echivalenți mulțimii de atribute X.

Notăm cu $\bar{E}_F=\{E_F(X)|\ X \subseteq R\ \&\ E_F(X) \neq \varnothing\}$, adică $\bar{E}_F$ este mulțimea tuturor claselor de echivalență nevide, în care este partiționată mulțimea de dependențe funcționale F.

Exemplul 3.17. Fie $F=\{AB \rightarrow C, AC \rightarrow D, AD \rightarrow B, C \rightarrow B\}$. Atunci $(AB)^+ = (AC)^+ = (AD)^+ = ABCD$ şi $C^+=BC$. Deci $AB \leftrightarrow AC$, $AD \leftrightarrow AC$, $AB \leftrightarrow AD$, adică $\bar{E}_F=\{E_F(AB)=\{AB \rightarrow C, AC \rightarrow D, AD \rightarrow B\}, E_F(C)=\{C \rightarrow B\}\}$.

Următoarea lemă arată corelația dintre structurile a două acoperiri nonredundante.

Lema 3.3. Fie F şi G două mulțimi de dependențe funcționale nonredundante echivalente asupra schemei R. Fie $X \rightarrow Y$ o dependență în F. Atunci în G există o dependență $V \rightarrow W$, unde $X \leftrightarrow V$ în raport cu F (şi în raport cu G).

Demonstrație. Din faptul că $F \equiv G$ urmează $G|=X \rightarrow Y$. Atunci există derivația H a dependenței funcționale $X \rightarrow Y$, în raport cu G. Considerăm toate dependențele utilizate în construirea derivației H, adică mulțimea U(H). În acelaşi timp, orice dependență $V \rightarrow W$ din U(H) se deduce din F. Fie H^1 este derivația dependenței $V \rightarrow W$ în F. Trebuie să existe în U(H) o dependență $V \rightarrow W$ pentru deducerea căreia se utilizează dependența $X \rightarrow Y$, adică $X \rightarrow Y \in U(H^1)$. În caz contrar pentru dependența $X \rightarrow Y$ va exista o derivație H^{11} asupra $F \setminus \{X \rightarrow Y\}$ şi, prin urmare, $X \rightarrow Y$ va fi redundantă în F ce contrazice ipotezei că F este o mulțime nonredundantă.

Întrucât $X \rightarrow Y \in U(H^1)$, conform consecinței 3.2, $F|=V \rightarrow X$. Dar $V \rightarrow W \in U(H)$ şi atunci $G|=X \rightarrow V$. Prin urmare, $X \leftrightarrow V$.

Lema de mai sus poate fi parafrazată în felul următor. În două acoperiri nonredundante F şi G pentru orice dependență din F există o dependență în G ce are partea stângă echivalentă celei din F. Prin urmare, mulțimile nonredundante echivalente au acelaşi număr de clase de echivalență.

Exemplul 3.18. Mulțimile $F=\{A\to BC, B\to A, AD\to E\}$ şi $G=\{A\to ABC, B\to A, BD\to E\}$ sunt nonredundante şi echivalente. Să observăm că $A\leftrightarrow A$, $B\leftrightarrow B$, $AD\leftrightarrow BD$ şi $A\leftrightarrow B$. Deci $\bar{E}_F=\{E_F(A) = \{A\to BC, B\to A\}, E_F(AD) = \{AD\to E\}\}$ şi $\bar{E}_G =\{E_G(A) = \{A\to ABC, B\to A\}, E_G(BD) = \{BD\to E\}\}$.

3.6.6. Acoperiri minimale

Definiția 3.22. Mulțimea de dependențe funcționale F este *minimală*, dacă nu există o mulțime echivalentă ei cu mai puține dependențe funcționale.

Este evident că orice mulțime minimală de dependențe funcționale este şi nonredundantă. Afirmația inversă nu este corectă.

Exemplul 3.19. Mulțimea $F=\{A\to B, A\to C\}$ este nonredundantă, dar nu este minimală, fiindcă $G=\{A\to BC\}$ este acoperire pentru F şi are o singură dependență.

Fie $e_F(X)$ este mulțimea părților stângi ale dependențelor ce formează clasa de echivalență $E_F(X)$. Atunci are loc

Lema 3.4. Fie F o mulțime nonredundantă de dependențe funcționale, X determinantul unei dependențe din F şi Y o mulțime de atribute echivalentă lui X (adică $Y\leftrightarrow X$ în raport cu F). Atunci există în $e_F(X)$ o mulțime Z, încât $(F \setminus E_F(X))|{=}Y\to Z$.

Demonstrație. Dacă $Y\in e_F(X)$, atunci lema e demonstrată, fiindcă $Y\to Y$ urmează din orice mulțime de dependențe funcționale. Fie $Y\notin e_F(X)$. Întrucât $Y\to Z$ pentru orice Z din $e_F(X)$, atunci există derivația $H=<Y_0, Y_1, ..., Y_m>$, unde $Y_0=Y$ şi $Z \subseteq Y_m$. Dacă în construirea lui H nu s-a utilizat nici o dependență din $E_F(X)$, atunci lema e demonstrată. Presupunem că pentru construirea derivației H s-a utilizat dependența $V\to W\in E_F(X)$ şi fie $V\to W$ e prima dependență din $E_F(X)$ utilizată în H. Atunci în H există un Y_i încât $V\subseteq Y_i$ dar $W\not\subseteq Y_i$. Dar $Y\to Y_i\in(F \setminus E_F(X))^+$ şi, conform reflexivității, $Y_i\to V$ are loc în orice mulțime de dependențe. Aplicând regula tranzitivității asupra ultimelor dependențe, obținem că $(F \setminus E_F(X))|{=}Y\to V$, adică Z=V.

Lema 3.5. Fie F şi G două mulțimi nonredundante echivalente de dependențe funcționale. Fie X e determinantul unei dependențe din F şi Y o mulțime de atribute, unde $Y\leftrightarrow X$. Dacă $Y\to Z\in(F \setminus E_F(X))^+$, atunci $Y\to Z\in(G \setminus E_G(X))^+$.

Demonstrație. Întrucât $Y\to Z\in(F \setminus E_F(X))^+$, atunci există derivația $H=<Y_0, Y_1, ..., Y_m>$, pentru $Y\to Z$ în raport cu $F \setminus E_F(X)$. Fie $V\to W$ o dependență utilizată în construirea lui H. Din $F\equiv G$ urmează $V\to W\in G^+$.

Să arătăm că în derivația dependenței $V\to W$ în raport cu G nu sunt utilizate dependențe din $E_G(X)$. Presupunem contrariul: pentru derivarea $V\to W$ este utilizată dependența $T\to S$ din $E_F(X)$. Atunci, conform lemei 3.3, $Y\leftrightarrow T$, iar din consecința 3.2 $V\to T\in G^+$. Din $V\to T\in G^+$ şi $T\to Y\in G^+$ urmează $V\to Y\in G^+$. Însă $Y\to V\in G^+$ şi atunci obținem că $Y\leftrightarrow V$ în raport cu F. Contrazicere. Deci, orice dependență utilizată în derivația lui $Y\to Z$ în raport cu $F\setminus E_F(X)$ se deduce din $G\setminus E_G(H)$.

Deci, derivația dependenței $Y \rightarrow Z$ va utiliza numai dependențe din $G \setminus E_G(H)$.

Teorema 3.7. O mulțime nonredundantă F este minimală, dacă şi numai dacă nici o clasă de echivalență $E_F(X)$ nu conține două dependențe diferite $X \rightarrow Y$ şi $V \rightarrow W$, unde $X \rightarrow V \in (F \setminus E_F(X))^+$.

Demonstrație. *Necesitatea.* Fie F este o mulțime minimală, şi presupunem contrariul: în clasa de echivalență $E_F(X)$ sunt două dependențe diferite $X \rightarrow Y$ şi $V \rightarrow W$ încât $X \rightarrow V \in (F \setminus E_F(X))^+$. Construim o mulțime de dependențe G, care se deosebeşte de F, prin aceea că în clasa de echivalență $E_G(X)$ dependențele $X \rightarrow Y$ şi $V \rightarrow W$ sunt substituite de $V \rightarrow YW$. Vom arăta ca $F \equiv G$. Pentru aceasta e suficient să verificăm dacă $X \rightarrow Y \in G^+$. Conform lemei 3.5 $X \rightarrow V \in (G \setminus E_G(X))^+$. Din $X \rightarrow V \in (G \setminus E_G(X))^+$ şi $V \rightarrow YW \in G^+$ urmează că $X \rightarrow YW \in G^+$. Deci $F \equiv G$, însă G conține cu o dependență mai puțin, ce contrazice ipotezei că F e o mulțime minimală de dependențe funcționale.

Suficiența. Vom arăta că, dacă orice clasă de echivalență a unei mulțimi F nu conține două dependențe diferite, încât părțile stângi să se determine funcțional în afara propriei clase de echivalență, atunci F este minimală. Să demonstrăm că nu există o mulțime nonredundantă G şi $G \equiv F$, încât careva clasă de echivalență din G să conțină mai puține dependențe decât clasa corespunzătoare din F.

Presupunem contrariul: există o mulțime nonredundantă G şi $G \equiv F$, iar clasa de echivalență $E_G(X)$ conține mai puține dependențe decât clasa $E_F(X)$.

Să evidențiem dependențele acestor două clase de echivalență, unde $m<n$ (vezi fig.3.10).

$E_F(X)$	$E_G(X)$
$X_1 \rightarrow Y_1$	$V_1 \rightarrow W_1$
$X_2 \rightarrow Y_2$	$V_2 \rightarrow W_2$
…	…
$X_n \rightarrow Y_n$	$V_m \rightarrow W_m$

Fig.3.10.

În corespundere cu lema 3.4 pentru orice $X_j \in e_F(X)$ există în $e_G(X)$ un determinant V_k şi $X_j \rightarrow V_k \in (G \setminus E_G(X))^+$. Apelând la lema 3.5. obținem $X_j \rightarrow V_k \in (F \setminus E_F(X))^+$. Întrucât $n>m$, atunci se vor găsi în $e_F(X)$ cel puțin două mulțimi X_j şi X_l, unde $j \neq l$, ce satisfac $X_j \rightarrow V_k \in (F \setminus E_F(X))^+$ şi $X_l \rightarrow V_k \in (F \setminus E_F(X))^+$. La rândul său, în $e_F(X)$ există un determinant X_h şi $V_k \rightarrow X_h \in (F \setminus E_F(X))^+$. Considerăm două cazuri posibile: $h \neq j$ sau $h \neq l$. Dacă $h \neq j$, atunci $X_j \rightarrow V_k \in (F \setminus E_F(X))^+$ şi $V_k \rightarrow X_h \in (F \setminus E_F(X))^+$ implică $X_j \rightarrow X_h \in (F \setminus E_F(X))^+$. Dacă, însă, $h \neq l$, atunci obținem $V_l \rightarrow X_h \in (F \setminus E_F(X))^+$.

În ambele cazuri, clasa de echivalență $E_F(X)$ va conține două dependențe diferite, părțile stângi ale cărora se determină funcțional în afara clasei de echivalență examinată. Contrazicere.

Consecința 3.3. Dacă F şi G sunt mulțimi minimale echivalente, atunci clasele de echivalență corespunzătoare conțin acelaşi număr de dependențe funcționale.

Consecința 3.4. Dacă F şi G sunt două mulțimi minimale echivalente, atunci pentru orice determinant $X_j \in e_F(X)$ există un singur determinant V_k în $e_G(X)$ pentru care au loc $X_j \rightarrow V_k \in (F \backslash E_F(X))^+$ şi $V_k \rightarrow X_j \in (F \backslash E_F(X))^+$.

Remarcă. Existența corespondenței biunivoce, indicate în consecința 3.4, permite substituirea unor părți stângi ale mulțimii minimale cu părți stângi ale altei acoperiri minimale, neafectând echivalența mulțimilor. Mai mult ca atât, mulțimea nouă de dependențe funcționale va continua să fie minimală.

În teorema de mai sus se afirmă că, dacă o mulțime nonredunantă G are două dependențe $X \rightarrow Y$ şi $V \rightarrow W$ pentru care $X \leftrightarrow V$ şi $(G \setminus E_G(X)) \models X \rightarrow V$, atunci G nu este minimală. Aceste două dependențe pot fi substituite cu altă dependență $V \rightarrow YW$. În rezultat obținem o mulțime echivalentă de dependențe funcționale ce conține cu o dependență mai puțin.

Acest proces este pus la baza următorului algoritm de minimizare a unei mulțimi de dependențe funcționale.

Algoritmul MINIMIZE (f,g)

Intrare: F – o mulțime de dependențe funcționale.
Ieşire: G – o mulțime minimală de dependențe funcționale.

```
begin
      NONREDUN (F, G);
      get Ē_G;
      for all E_G(X) în Ē_G
         for all X→Y in E_G(X)
            for all V→W≠X→Y in E_G(X)
               if MEMBERSHIP (G \ E_G(X), X→V) then G:= G \ {X→Y,
               V→W} ∪ {V→YW};
      return (G);
end
```

Exemplul 3.20. Fie F={AB→D, AB→C, AC→D, AD→B, C→B}. Să construim acoperirea minimală a mulțimii F.

Nu e greu de observat că, dacă examinăm dependențele funcționale din F de la stânga la dreapta, dependența funcțională AB→D e redundantă în F. Într-adevăr, $(AB)^+$ în raport cu F\{AB→D} este ABCD. Deci $(F \backslash \{AB \rightarrow D\}) \models AB \rightarrow D$. Am obținut mulțimea nonredundantă G={AB→C, AC→D, AD→B, C→B}.

Să partiționăm G în clase de echivalență. Pentru aceasta construim închiderile determinanților tuturor dependențelor din G. Dependențele ce au închideri ale determinanților egale fac parte din aceeaşi clasă de echivalență. Aşadar,

$(AB)^+$=ABCD,
$(AC)^+$=ABCD,
$(AD)^+$=ABCD,
C^+=BC.

Deci, mulțimea G conține următoarele două clase de echivalență $G=\{E_G(AB)=\{AB \rightarrow C, AC \rightarrow D, AD \rightarrow B\}, E_G(C)=\{C \rightarrow B\}\}$.

Întrucât clasa $E_G(C)$ conține o singură dependență, se examinează numai clasa de echivalență $E_G(AB)$. Nu e greu de verificat că în $E_G(AB)$ sunt două dependențe $AC \rightarrow D$, $AB \rightarrow C$ şi $(G \setminus E_G(AB)) \mid = AC \rightarrow AB$. Atunci în clasa de echivalență $E_G(AB)$ aceste două dependențe se substituie cu $AB \rightarrow CD$.

Am obținut mulțimea de dependențe $G=\{E_G(AB)=\{AB \rightarrow CD, AD \rightarrow B\}, E_G(C)=\{C \rightarrow B\}\}$. În clasa de echivalența $E_G(AB)$ nu se mai găsesc dependențe determinații cărora se determină funcțional în afara clasei $E_G(AB)$. Prin urmare, mulțimea $G=\{AB \rightarrow CD, AD \rightarrow B, C \rightarrow B\}$ este o acoperire minimală a mulțimii F.

3.6.7. Acoperiri optimale

Mulțimea de dependențe funcționale F poate fi estimată după numărul de atribute (inclusiv repetate) antrenate de dependențele funcționale din F. De pildă, mulțimea $F=\{AB \rightarrow C, C \rightarrow B\}$ are aritatea cinci.

Definiția 3.23. Mulțimea de dependențe funcționale F se numeşte *optimală*, dacă nu există o mulțime echivalentă ei cu o aritate mai mică.

Exemplul 3.21. Mulțimea $F=\{ABC \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ nu este acoperire optimală, fiindcă mulțimea $G=\{AD \rightarrow E, BC \rightarrow D, D \rightarrow BC\}$ are aritatea mai mică decât F şi $G \equiv F$. Mulțimea G este optimală.

Trebuie menționat că problema construirii unei acoperiri optimale aparține clasei de probleme NP-complete, pentru care încă nu au fost găsiți algoritmi polinomiali.

Teorema 3.8. Mulțimea optimală este minimală şi redusă.
Demonstrarea acestei afirmații se lasă în calitate de exercițiu.

3.7. Exerciții

3.1. Să se aducă un exemplu de două atribute ce se găsesc într-o dependență funcțională şi un exemplu de două atribute ce nu sunt funcțional dependente.

r	A	B	C	D
	a_1	b_1	c_1	d_1
	a_1	b_1	c_2	d_2
	a_2	b_1	c_1	d_3
	a_2	b_1	c_3	d_4

Fig.3.11.

3.2. Să se găsească dependențele funcționale valide în relația r din fig.3.11.

3.3. Fie r e definită pe mulțimea ABC. Să se aducă o extensie a relației r ce ar satisface dependența funcțională $A \rightarrow B$ şi nu ar satisface dependența $C \rightarrow A$.

3.4. Să se arate că $\{WX \rightarrow Y\} \mid - X \rightarrow Y$.

3.5. Fie relația r(R) şi V, W, X, Y, $Z \subseteq R$. Să se demonstreze sau să se combată următoarele reguli de inferență.

(a) $\{X \to Y, Z \to W\} \mid\!- XZ \to YW$;

(b) $\{XY \to Z, Z \to X\} \mid\!- Z \to Y$;

(c) $\{X \to Y, Y \to Z\} \mid\!- X \to YZ$;

(d) $\{ZV \to W, W \to XV, V \to Y\} \mid\!- ZV \to XY$;

(e) $\{X \to Y, W \to Z\} \mid\!- X \to Z$, unde $W \subseteq Y$.

3.6. Să se arate că regulile DF1, DF2 şi DF6 sunt independente, adică nici una din ele nu se deduce din celelalte.

3.7. Să se arate că pentru orice mulțime de dependențe funcționale F are loc egalitatea $F^+ \equiv (F^+)^+$.

3.8. Fie F o mulțime de dependențe funcționale asupra schemei R. Care este F^+, dacă $F=\varnothing$?

3.9. Fie mulțimea $F=\{AB \to C, C \to B\}$ definită asupra atributelor ABC. Să se găsească F^+.

3.10. Să se demonstreze că sistemul de reguli DF1, DF3, DF4 şi DF5 este complet. Sunt aceste reguli independente?

3.11. Fie $F=\{AB \to C, B \to D, CD \to E, CE \to GH, G \to A\}$.

(a) Să se construiască consecutivitatea de derivare a dependenței $AB \to E$ din F.

(b) Să se construiască consecutivitatea de derivare a dependenței $AB \to G$ din F, utilizând axiomele Armstrong.

(c) Să se construiască RAP–consecutivitatea de derivare a dependenței $AB \to G$ din F.

(d) Să se construiască DDA-graful de derivare a dependenței $AB \to G$ din F.

3.12. Să se arate că mulțimile $\{AB \to E, CD \to F, A \to C, B \to D, C \to A, D \to B, F \to AD\}$ şi $G=\{A \to C, B \to D, C \to A, D \to B, F \to AD, CD \to EF\}$ sunt echivalente.

3.13. Să se găsească mulțimea claselor de echivalență $\bar{E}_G$, dacă $G=\{AB \to EF, A \to C, B \to D, C \to A, D \to B, F \to AD\}$.

3.14. Să se construiască o acoperire nonredundantă a mulțimii de dependențe funcționale $F=\{A \to C, AB \to C, C \to DI, CD \to I, EC \to AB, EI \to G\}$.

3.15. Să se construiască o mulțime de dependențe funcționale în care o dependență funcțională $X \to Y$ are toate atributele redundante.

3.16. Să se construiască două mulțimi de dependențe funcționale F şi G, unde F e o acoperire nonredundantă a mulțimii G, dar conține un număr mai mare de dependențe decât G.

3.17. Fie F mulțimea tuturor dependențelor posibile asupra schemei $R=A_1...A_n$, în afară de $\varnothing \to X$. Să se găsească o acoperire nonredundantă a mulțimii F.

3.18. Să se găsească două mulțimi nonredundante echivalente cu un număr diferit de dependențe.

3.19. Fie $F = \{A \to B, B \to C, C \to A\}$ şi $G = \{A \to BC, B \to A, C \to A\}$.

(a) Să se arate că mulțimile F şi G sunt echivalente.

(b) Să se găsească o acoperire redusă a mulțimii G.

3.20. Fie $G=\{AF \to C, C \to D, D \to BE, B \to CE, CE \to A\}$. Să se găsească o acoperire canonică a mulțimii de dependențe G.

3.21. Să se găsească acoperire minimală a mulțimii $G=\{AB \to C, BC \to D, BE \to C, CD \to B, CE \to AF, CF \to BD, C \to A, D \to EF\}$.

3.22. Să se construiască o acoperire optimală a mulțimii $G=\{A \to BC, BC \to A, ABD \to EF, BCD \to EF\}$.