Capitolul 6

BAZE DE DATE ACICLICE

Concluzionând cele descrise în secțiunile precedente, o schemă “bună” a bazei de date trebuie să posede mai multe calități dezirabile. Printre aceste calități putem menționa, în primul rând, formele normale, proprietatea joncțiunii fără pierderi şi conservarea dependențelor.

Însă, asupra schemei bazei de date mai pot fi definite nişte constrângeri sintactice cum ar fi, spre exemplu, aciclicitatea. Se cunosc diferite tipuri de aciclicitate. Similar unei ierarhii de forme normale ale schemelor, fiecare formă fiind mai restrictivă decât predecesoarea, există şi o ierarhie de tipuri de aciclicități. După cum se ştie, proiectantul bazei de date trebuie să țină cont că, dacă schema relațională nu se găseşte în forma normală corespunzătoare, atunci pot apărea diverse probleme de actualizare a bazei de date. De asemenea, de competența proiectantului ține şi selectarea gradului de aciclicitate în care doreşte ca schema să fie proiectată.

Schemele aciclice se bucură de o serie de proprietăți. Cu cât gradul de aciclicitate este mai înalt, cu atât mai “bună” este schema. Mai mult decât atât, unii algoritmi, ce au o complexitate exponențială asupra schemelor ciclice, asupra schemelor aciclice, sunt polinomiali.

Schemele aciclice ale bazelor de date pot fi caracterizate în diferite moduri. În primul rând, definiția de aciclicitate poate fi formulată prin forme echivalente. Toate aceste forme se bazează pe reprezentarea schemelor bazelor de date cu ajutorul hipergrafurilor. Unele definiții de aciclicități se aduc, utilizând componentele hipergrafurilor în timp ce altele sunt bazate pe grafuri ordinare construite din hipergrafuri.

Multe din proprietățile schemelor aciclice pot fi concepute în calitate de caracteristici, în sens că schema are o proprietate particulară, dacă şi numai dacă schema este aciclică. O parte din proprietăți sunt strâns legate de procesarea interpretărilor la baza de date.

6.1. Scheme hipergrafuri

Întrucât schema bazei de date este o mulțime de scheme relaționale, e foarte comod de a asocia schemei bazei de date un hipergraf.

Vom aduce noțiunea de hipergraf. Hipergraful este analogic grafului ordinar neorientat, cu excepția că o muchie a lui nu uneşte numai două noduri, ci o mulțime arbitrară de noduri.

FURNIZOR	CONTRACT	DATĂ

FURNIZOR	ARTICOL	PREȚ

FURNIZOR	ARTICOL	CONTRACT

Fig.6.1. Schema bazei de date Db={FURNIZOR CONTRACT DATĂ, FURNIZOR ARTICOL PREȚ, FURNIZOR ARTICOL CONTRACT}

Definiția 6.1. Hipergraful H este o pereche (N, E), unde N este o mulțime finită de noduri şi E o mulțime de muchii (sau hipermuchii), care sunt submulțimi nevide ale mulțimii de noduri N.

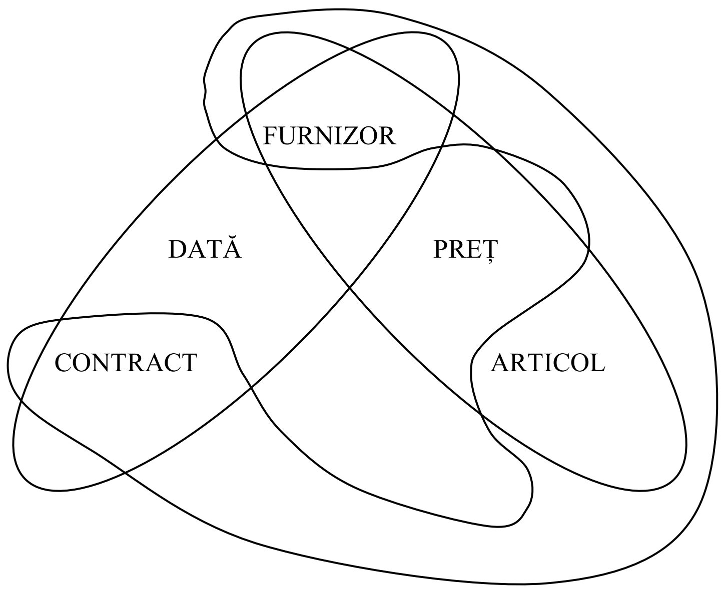

Fig.6.2. Hipergraful schemei bazei de date din fig.6.1

Mai departe vom identifica un hipergraf numai prin menționarea muchiilor sale şi implicit vom presupune că mulțimea de noduri este exact mulțimea nodurilor ce aparțin tuturor muchiilor.

Schemei bazei de date îi vom pune în corespondență un hipergraf în felul următor. Mulțimea de atribute U ce formează mulțimea universală este mulțimea de noduri, iar fiecare schemă relațională din schema bazei de date reprezintă o muchie ce include nodurile notate cu atributele din schema relațională. Hipergraful din fig.6.2 corespunde schemei bazei de date din fig.6.1.

Considerăm două scheme ale bazelor de date din fig.6.3 şi fig.6.4, fiecare conținând trei scheme relaționale. Unica diferență dintre aceste scheme este că a doua schemă a bazei de date are atributul D în ultima schemă relațională, în timp ce prima schemă a bazei de date conține atributul E. Cu toate că această diferență la prima vedere pare neesențială, în realitate, schema din figura 6.3 este aciclică, iar cea din fig.6.4 e ciclică. Mai departe vom vedea că prima schemă posedă o serie de priorități dezirabile, dar a doua - nu. Pentru o vizualizare a faptului că a doua schemă este ciclică considerăm hipergrafurile corespunzătoare din fig.6.5.

A	B	C

B	C	D

A	E

Fig.6.3. Schema Db_1={ABC, BCD, AE}

A	B	C

B	C	D

A	D

Fig.6.4. Schema Db_2={ABC, BCD, AD}

Nu vom aduce aici definiția aciclicității, vom spune doar că primul hipergraf este aciclic, iar al doilea - ciclic. Adică vom spune că schema Db_1 este aciclică şi Db_2 este ciclică.

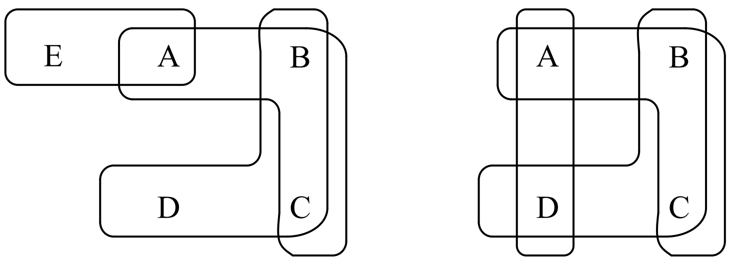

Fig.6.5. Hipergrafurile schemelor Db_1 şi Db_2

În acest capitol, nu vom examina toate proprietățile pe care le posedă o schemă aciclică a bazelor de date. Aceste proprietăți au fost studiate de mulți cercetători în diferit context. Cel mai remarcabil fapt este însă că toate aceste proprietăți sunt echivalente (în sens că ele sunt echivalente aciclicității).

6.2. Algoritmul Graham

Există un simplu algoritm de determinare a aciclicității schemei bazei de date. Din considerente pedagogice, expunerea formală a aciclicității o lăsăm pentru secțiunea următoare.

Algoritmul Graham de determinare a aciclicității constă în reducerea pas cu pas a hipergrafului, conform a două reguli până nici una din reguli nu mai poate fi aplicată asupra hipergrafului, reprezentând schema bazei de date.

Dacă în urma aplicării regulilor obținem un hipergraf vid, atunci schema bazei de date este aciclică, în caz contrar este ciclică.

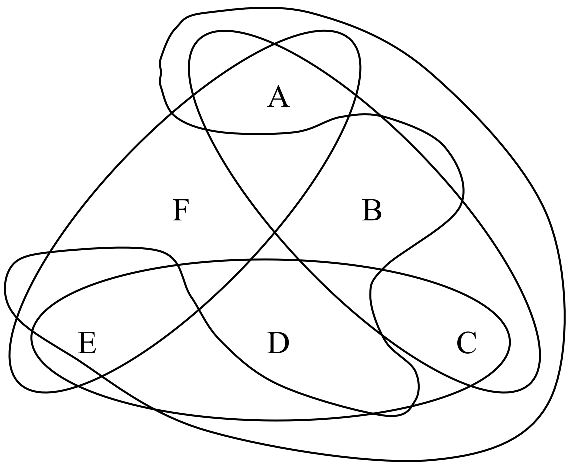

Fig.6.6. Hipergraful schemei Db={ABC,EDC,AEF,ACE}

Fie hipergraful H=(N, E). Regulile de reducere sunt următoarele.

EM (eliminare muchie). Muchia E_i se elimină din E, dacă există o muchie E_j, $i \neq j$, încât $E_i \subseteq E_j$.

EN (eliminare nod). Dacă nodul n_i aparține numai unei singure muchii, el este eliminat din N, deci şi din muchia din care face parte.

Exemplul 6.1. Considerăm hipergraful din fig.6.6 şi să verificăm, aplicând algoritmul de mai sus, dacă el este aciclic sau nu.

Pentru comoditate, vom reprezenta fiecare muchie a hipergrafului prin nodurile sale amplasate pe o linie, nodurile comune ale muchiilor fiind puse unul sub altul.

Deci algoritmul începe cu considerarea hipergrafului:

A	B	C			
		C	D	E	
A				E	F
A		C		E	

Se aplică regulile EM şi EN, până nu mai pot fi făcute modificări asupra hipergrafului.

Aplicăm regula EN, pentru a înlătura nodurile izolate (ce aparțin unei singure muchii). În exemplul nostru, se înlătură nodurile B, D şi F, fiind izolate. Au rămas:

A	C	
	C	E
A		E
A	C	E

Cu ajutorul regulii EM, eliminăm muchiile nodurile cărora se găsesc în alte muchii. Aşadar, prima muchie AC se conține în ultima ACE şi, înlăturând-o pe prima, obținem:

	C	E
A		E
A	C	E

Similar, din aceeaşi cauză, înlăturăm prima, CE, şi a doua, AE, muchii. Atunci obținem un hipergraf format dintr-o singură muchie:

A	C	E

Apelând din nou la regula EN, sunt eliminate nodurile A, C şi E.

Am obținut o mulțime vidă. Deci schema bazei de date reprezentată de hipergraful din fig.6.6 este aciclică.

Teorema 6.1. Un hipergraf (sau o schemă) este aciclică, dacă aplicând algoritmul Graham obținem o mulțime vidă.

În legătură că demonstrările unor teoreme sunt destul de complexe şi necesită cunoştințe suplimentare ce depăşesc cadrul acestei lucrări, demonstrările nu vor fi aduse. Teoremele vor fi pur şi simplu numai formulate.

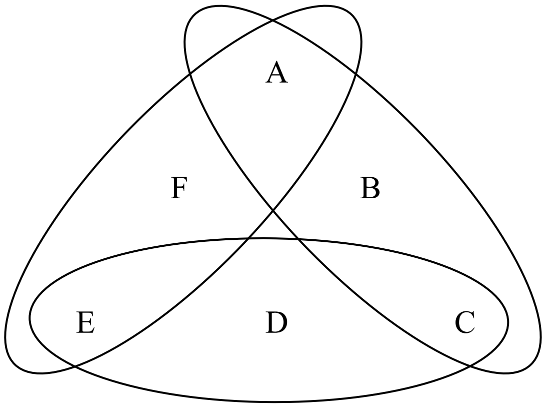

Fig.6.7. Hipergraful schemei Db={ABC, CDE, AEF}

Exemplul 6.2. Pentru hipergraful din fig.6.7, algoritmul Graham nu produce o mulțime vidă. Deci schema reprezentată de el este ciclică.

Într-adevăr, algoritmul începe, considerând următorul hipergraf:

A	B	C			
		C	D	E	
A				E	F

După înlăturarea nodurilor izolate B, D şi F obținem:

A	C	
	C	E
A		E

Aici aplicarea regulilor EM şi EN nu mai e posibilă şi, prin urmare, hipergraful considerat este ciclic. Deci ciclică este şi schema bazei de date asociată acestui hipergraf.

Acest exemplu este instructiv, din punctul de vedere, că el ne demonstrează că nu orice subhipergraf al unui hipergraf aciclic este aciclic. Hipergraful din fig.6.7 este un subhipergraf al hipergrafului din fig.6.6.

Prin subhipergraf al unui hipergraf vom înţelege o submulţime de muchii şi noduri al hipergrafului. Acest fenomen contraintuitiv nu are loc pentru grafurile ordinare, adică nu e posibil ca un subgraf al unui graf ordinar aciclic să fie ciclic.

În secţiunile următoare, vor fi considerate alte tipuri de aciclicitate pentru hipergrafuri, în care fenomenul de mai sus nu are loc.

6.3. Consistenţe

Vom examina o proprietate a bazei de date ce este echivalentă aciclicităţii.

Fie schema Db={R_1, ..., R_m} a bazei de date db={r_1,...,r_m}. În secţiunile precedente spuneam că verificarea, dacă o schemă a bazei de date posedă proprietatea joncţiunii fără pierderi, este o problemă laborioasă.

Definiţia 6.2. Vom spune că baza de date db={r_1, ..., r_m} este *consistentă,* dacă posedă proprietatea joncţiunii fără pierderi şi *consistentă câte două*, dacă orice pereche r_i, r_j de relaţii din db posedă proprietatea joncţiunii fără pierderi.

r_1	A	B	C
	a_0	b_0	c_1
	a_1	b_0	c_1
	a_2	b_3	c_4

r_2	B	C	D
	b_0	c_1	d_1
	b_3	c_4	d_5

r_3	A	D
	a_0	d_1
	a_1	d_1
	a_2	d_5

Fig.6.8. Baza de date db = {r_1(ABC), r_2(BCD), r_3(AD)}

Exemplul 6.3. Baza de date din fig.6.8 este joncţionabilă fără pierderi fiindcă relaţiile r_1, r_2, r_3 sunt proiecţiile relaţiei universale din fig.6.9. E uşor de verificat că şi orice două relaţii din db = {r_1, r_2, r_3} sunt joncţionabile fără pierderi.

r	A	B	C	D
	a_0	b_0	c_1	d_1
	a_1	b_0	c_1	d_1
	a_2	b_3	c_4	d_5

Fig.6.9. Relaţia universală r(ABCD)

Exemplul 6.4. Să observăm că baza de date din fig.6.10 nu este joncționabilă fără pierderi.

Într-adevăr tuplurile <a_0 b_0 c_0> şi <b_0 c_0 d_0> din r_1 şi r_2 sunt joncționabile, formând tuplul < a_0 b_0 c_0 d_0>. Dar, proiectând tuplul < a_0 b_0 c_0 d_0> pe mulțimea de atribute AD, nu poate fi obținut nici un tuplu din relația r_3. Orice două relații din fig.6.10 sunt joncționabile fără pierderi, dar baza de date în întregime nu este joncționabilă fără pierderi.

Aceasta are loc, fiindcă schema bazei de date Db={ABC, BCD, AD} este ciclică.

r_1	A	B	C
	a_0	b_0	c_0
	a_1	b_1	c_1

r_2	B	C	D
	b_0	c_0	d_0
	b_1	c_1	d_1

r_3	A	D
	a_0	d_1
	a_1	d_0

Fig.6.10. Baza de date r = {r_1, r_2, r_3}

Teorema 6.2. Dacă schema este aciclică, atunci baza de date este consistentă, dacă şi numai dacă e consistentă câte două.

Teorema 6.3. Dacă schema este ciclică, atunci există o bază de date consistentă câte două, dar care nu e consistentă.

Deci, verificarea dacă o bază de date este joncționabilă fără pierderi, în cazul când schema ei este aciclică, constituie o procedură simplă. Se verifică dacă relațiile ei sunt joncționabile două câte două. Această procedură are complexitate polinomială, spre deosebire de cazul când schema este ciclică.

6.4. Program semijoncțiune de reducere completă

Să examinăm o problemă legată de altă proprietate echivalentă noțiunii de aciclicitate. Fie că avem o bază de date distribuită, geografic plasată în mai multe oraşe, câte o relație în fiecare oraş.

r_1	A	B	C	D	E	F	G
t_1	a_0	b_0	c_1	d_2	e_3	f_4	g_5
t_2	a_1	b_3	c_9	d_0	e_6	f_3	g_{17}
t_3	a_2	b_1	c_{17}	d_4	e_{19}	f_2	g_8

Fig.6.11. Relația r_1 în oraşul O_1

r_2	A	B	H	I	K	L	M
t_1	a_0	b_0	h_{101}	i_{101}	k_{103}	l_{104}	m_{105}
t_2	a_1	b_3	h_{201}	i_{102}	k_{203}	l_{204}	m_{205}

t_3	a_2	b_1	h_{14}	i_{14}	k_3	l_6	m_{47}

Fig. 6.12. Relația r_2 în orașul O_2

De exemplu, fie relația din fig.6.11 se găseşte în oraşul O_1, iar cea din fig.6.12 în oraşul O_2.

Presupunem că fiecare din relațiile r_1 şi r_2 conțin multe tupluri (sunt prezentate doar câteva). Pentru a obține răspuns la o interpelare ce antrenează atribute din ambele relații, se poate întâmpla că transmiterea datelor între oraşe costă mult mai scump, decât prelucrarea datelor în fiecare punct aparte. Deci, vom încerca să soluționăm problema de minimizare a volumului de date, transferate de la un punct la altul.

Transmiterea relațiilor într-un punct, apoi efectuarea operațiilor necesare pentru extragerea răspunsului la interpelare, apoi returnarea răspunsului poate fi foarte ineficientă. La joncțiunea relațiilor pot participa doar o parte de tupluri, celelalte fiind nerelevante interpretării date. În cazul relațiilor noastre la joncțiune participă doar tuplul t_1 din r_1 şi tuplurile t_1 şi t_2 din r_2, rezultatul fiind reprezentat în fig.6.13.

r	A	B	C	D	E	F	G	H	I	K	L	M
	a_0	b_0	c_1	d_2	e_3	f_4	g_5	h_{101}	i_{102}	k_{103}	l_{104}	m_{105}
	a_0	b_0	c_1	d_2	e_3	f_4	g_5	h_{201}	i_{202}	k_{203}	l_{204}	m_{205}

Fig. 6.13. Joncțiunea relațiilor r_1 şi r_2

Pentru soluționarea acestei probleme se propune aşa numita strategie a semijoncțiunii. Vom descrie lucrul acestei strategii pentru exemplul nostru.

Pasul 1. Se taie proiecția relației r_2 din oraşul O_2 asupra atributelor AB (acestea sunt atributele comune pentru relațiile din oraşele O_1 şi O_2 şi se transmite în oraşul O_1. Deci, se transmite un singur tuplu $<a_0\ b_0>$.
Pasul 2. Se retează relația r_1 din O_1, eliminând din r_1 acele tupluri ce nu se joncționează cu proiecția pe AB transmisă din oraşul O_2. (Să se observe că rezultatul obținut nu este altceva decât semijoncțiunea r_1 şi r_2, adică $r_1 \leftarrow r_1 |\!\times r_2$.
Pasul 3. Din oraşul O_1 relația obținută se transmite în oraşul O_2, unde se produce joncțiunea ei cu relația r_2.

Fie bd = $\{r_1, ..., r_m\}$ o bază de date distribuită şi fie că către această bază de date este adresată o interpelare . E de dorit ca din fiecare relație să fie înlăturate tuplurile ce nu participă la joncțiunea $r_1 |\times| ... |\times| r_m$.

Definiția 6.3. Vom numi *reducere completă* a relației r_i mulțimea tuturor tuplurilor din r_i ce participă la joncțiunea $|\times|$(bd) = $r_1|\times|...|\times|r_m$. Orice consecutivitate de atribuiri $r_i \leftarrow r_i |\!\times s_j$ se numeşte *program semijoncțiune*. Dacă pentru orice bază de date bd cu schema Bd şi pentru orice relație r_i, programul semijoncțiune pentru relația r_i produce reducerea completă a r_i , atunci acest program semijoncțiune se numeşte *program semijoncțiune de reducere completă* a schemei Bd.

Deci, dacă schema Bd posedă programul de reducere completă, atunci pot fi complet reduse relațiile $r_1,...,r_m$, înainte de a fi transmise într-un punct comun de prelucrare şi calculare a joncțiunii.

Însă într-un sistem distribuit concret, răspunsul la întrebarea, dacă e eficientă sau nu utilizarea unui sau altui program semijoncțiune, depinde de extensiile relațiilor aparte. Calculând $r(R)|\times|s(R)$ într-un sistem distribuit, poate fi mai puțin costisitor de transmis toată relația în punctul unde este alocată relația s, decât la început de transmis $\pi_{R\cap S}(s)$ în punctul de locație a relației r, apoi de transmis $r|\times s$ în punctul s. Pentru o bază de date concretă utilizarea unui program de reducere completă poate fi convenabilă, iar pentru alta - poate să nu fie eficientă.

Exemplul 6.5. Fie Db = {ABC, BCD, CD} schema bazei de date reprezentată în fig.6.14. Pentru schema Db există programe de reducere completă. Unul din ele poate fi:

$r_2 \leftarrow r_2 \; |\times \; r_1;$

$r_3 \leftarrow r_3 \; |\times \; r_2;$

$r_2 \leftarrow r_2 \; |\times \; r_3;$

$r_1 \leftarrow r_1 \; |\times \; r_2.$

Rezultatul acestui program e reprezentat în fig.6.15.

r_1	A	B	C
	a_7	b_4	c_6
	a_8	b_4	c_6
	a_7	b_5	c_6
	a_9	b_4	c_{11}
	a_8	b_5	c_{11}

r_2	B	C	D
	b_4	c_6	d_7
	b_5	c_6	d_7
	b_4	c_{11}	d_9
	b_5	c_{11}	d_9
	b_4	c_{12}	d_9

r_3	C	D
	c_6	d_7
	c_6	d_9
	c_8	d_9
	c_{11}	d_9

Fig.6.14. Baza de date db = $\{r_1, r_2, r_3\}$

Exemplul 6.6. Considerăm schema Db = {ABC, BCD, CE, DE} a bazei de date prezentate în fig.6.16. Pentru Db nu există un program de reducere completă

r_1^1	A	B	C
	a_7	b_4	c_6
	a_8	b_4	c_6
	a_7	b_5	c_6
	a_9	b_4	c_{11}
	a_8	b_5	c_{11}

r_2^1	B	C	D
	b_4	c_6	d_7
	b_5	c_6	d_7
	b_4	c_{11}	d_9
	b_5	c_{11}	d_9

r_3^1	C	D
	c_6	d_7

c_{11}	d_9

Fig.6.15. Reducerea completă $db^1 = \{r_1^1, r_2^1, r_3^1\}$ a bazei de date $db = \{r_1, r_2, r_3\}$ din fig.6.14.

r_1	A	B	C
	a_1	b_2	c_3
	a_7	b_8	c_9

r_2	B	C	D
	b_2	c_3	d_4
	b_8	c_9	d_{10}

r_3	C	E
	c_3	e_5
	c_9	e_{11}

r_4	D	E
	d_4	e_{11}
	d_{11}	e_5

Fig.6.16. Baza de date $db = \{r_1, r_2, r_3, r_4\}$

Teorema 6.4. Dacă schema bazei de date este aciclică, atunci există un program semijoncțiune ce reduce complet toate relațiile în baza de date.

Cu alte cuvinte, dacă schema este aciclică, atunci strategia semijoncțiunii este utilă. Însă, situația e completamente alta în cazul, dacă schema este ciclică.

Teorema 6.5. Dacă schema bazei de date este ciclică, atunci nu întotdeauna există un program semijoncțiune ce reduce complet toate relațiile în baza de date.

Remarcă. Deci, din teoremele 6.4 şi 6.5, conchidem că schema bazei de date este aciclică, dacă şi numai dacă există un program semijoncțiune de reducere completă a relațiilor în baza de date.

6.5. Expresii joncțiune monotone

Considerăm următorul scenariu. Utilizatorul a decis să joncționeze patru relații r_1, r_2, r_3, şi r_4. Poate să se întâmple următoarele. El joncționează la început r_1 şi r_2, $r_1|\times|r_2$, şi joncțiunea produce, spre exemplu, o relație cu o mie tupluri. Apoi el poate joncționa rezultatul cu r_3 , adică $r_1|\times|r_2|\times|r_3$, şi obține o relație, să zicem, cu un milion tupluri. În final joncțiunea relațiilor r_1, r_2, r_3, şi r_4, $r_1 |\times| r_2 |\times| r_3 |\times| r_4$, produce zece tupluri. Prin urmare, chiar dacă rezultatul conține doar câteva tupluri, rezultatele intermediare pot fi excesiv de mari.

Trebuie menționat că şi în cazul, când relațiile bazei de date sunt complet reductibile, o alegere nereuşită a joncțiunilor poate genera dimensiuni foarte mari ale rezultatelor intermediare.

Exemplul 6.7. Considerăm calcularea joncțiunii relațiilor $r_1|\times|r_2|\times|r_3$, ale bazei de date asupra schemei Db={ABC,BCD,CDE} din fig.6.17. Dacă începem cu calcularea $r_1 |\times| r_3$, atunci vom obține un rezultat intermediar cu 10 tupluri, în timp ce rezultatul final are 6 tupluri. Însă, dacă joncţiunea începe cu r_1 |x| r_2, atunci relaţia din rezultatul intermediar va conţine numai şase tupluri.

r_1	A	B	C
	a_1	b_3	c_5
	a_1	b_4	c_5
	a_2	b_3	c_5
	a_2	b_4	c_6

r_2	B	C	D
	b_3	c_5	d_7
	b_4	c_5	d_8
	b_3	c_5	d_9
	b_4	c_6	d_8

r_3	C	D	E
	c_5	d_7	e_{10}
	c_5	d_8	e_{10}
	c_5	d_9	e_{11}
	c_6	d_8	e_{11}

Fig.6.17. Baza de date cu schema Db={ABC, BCD, CDE}

Exemplul 6.8. Considerăm calcularea joncţiunii relaţiilor r_1|x|r_2|x|r_3|x|r_4, ale bazei de date asupra schemei Db={ABC, BCD, DE, CE}, reprezentată în fig.6.18. Joncţiunea oricăror perechi de relaţii produce un rezultat intermediar cu mai multe tupluri decât rezultatul final. Să menţionăm, însă, că această bază de date este complet reductibilă.

Pe noi ne interesează schemele bazelor de date, ale căror relaţii complet reductibile pot fi joncţionate într-o aşa consecutivitate de joncţiuni perechi încât numărul de tupluri în rezultatele intermediare să nu depăşească numărul de tupluri în rezultatul final. Mai mult decât atât, noi dorim să avem o astfel de consecutivitate, care ar avea această proprietate pentru toate bazele de date cu schema dată. În realitate noi vom cere satisfacerea unei condiţii mai stricte: de fiecare dată când se calculează joncţiunea, relaţiile ce participă în joncţiune trebuie să fie complet joncţionabile.

r_1	A	B	C
	a_1	b_2	c_3
	a_1	b_2	c_4
	a_1	b_2	c_5
	a_1	b_2	c_6
	a_1	b_2	c_7

r_2	B	C	D
	b_2	c_3	d_8
	b_2	c_4	d_8
	b_2	c_5	d_{11}
	b_2	c_5	d_{12}
	b_2	c_6	d_{16}
	b_2	c_7	d_{17}

r_3	D	E
	d_8	e_9
	d_8	e_{10}
	d_{11}	e_{13}
	d_{12}	e_{14}
	d_{16}	e_{15}
	d_{17}	e_{15}

r_4	C	E
	c_3	e_9
	c_4	e_{10}
	c_5	e_{14}
	c_6	e_{15}
	c_7	e_{15}

Fig.6.18. Baza de date db = $\{r_1, r_2, r_3, r_4\}$

Definiția 6.4. *Expresie joncțiune* este o expresie formată din scheme relaționale, simbolul "|x|" şi paranteze, între care orice joncțiune este binară (participă numai două scheme).

Exemplul 6.9. Dacă R_1, R_2, R_3 şi R_4 sunt scheme relaționale, atunci $((R_2 \,|x|\, R_3) \,|x|\, (R_1 \,|x|\, R_4))$ este o expresie joncțiune ce presupune joncțiunea relațiilor cu schemele R_2 şi R_3, joncțiunea relațiilor cu schemele R_1 şi R_4, şi apoi joncțiunea ambelor rezultate intermediare.

Fie θ o expresie joncțiune asupra tuturor schemelor din Db şi db o bază de date cu schema Db. Prin θ(db) vom subînțelege relația ce rezultă prin substituirea oricărei scheme R_i din θ cu r_i, unde $r_i \in$db şi r_i are schema R_i. De exemplu, dacă db=$\{r_1, r_2, r_3, r_4\}$ şi θ este expresia joncțiune $(R_2 |x| (R_3 \,|x|\, R_2))$, unde r_2 şi r_3 au schemele respectiv R_2 şi R_3, atunci θ(db) este relația $(r_2 \,|x|\, (r_3 \,|x|\, r_2))$, adică relația $r_2 \,|x|\, r_3$.

O subexpresie a expresiei joncțiune se defineşte în mod obişnuit.

Definiția 6.5. Fie θ o expresie joncțiune, conținând scheme relaționale din Db şi fie db o bază de date cu schema Db. Vom spune că θ este *monotonă în raport cu db*, dacă pentru orice subexpresie $(\theta_1 \,|x|\, \theta_2)$ din θ relațiile $\theta_1(r)$ şi $\theta_2(r)$ sunt consistente.

Intuitiv, θ este monotonă în raport cu db, dacă nici un tuplu nu este pierdut, când se execută careva joncțiune binară din θ (r).

Exemplul 6.10. Expresia $((R_2 \,|x|\, R_3) \,|x|\, (R_1 \,|x|\, R_4))$ este monotonă în raport cu db = $\{r_1, r_2, r_3, r_4\}$, unde r_i are schema R_i, $1 \le i \le 4$, dacă

(a) r_2 şi r_3 sunt consistente;
(b) r_1 şi r_4 sunt consistente;
(c) $(r_2 \,|x|\, r_3)$ şi $(r_1 \,|x|\, r_4)$ sunt consistente.

Definiția 6.6. Vom spune că expresia θ este monotonă, dacă ea este monotonă în raport cu orice bază de date consistentă câte două relații cu scheme din Db. Dacă θ include exact schemele din Db, atunci spunem că Db are *expresie joncțiune monotonă*.

Expresiile monotone asigură utilizarea eficientă a spațiului de memorie în timpul joncțiunilor, fiindcă nici o joncțiune intermediară nu are mai multe tupluri decât joncțiunea finală.

Teorema 6.6. O schemă a bazei de date este aciclică, dacă şi numai dacă există o expresie joncțiune monotonă.

6.6. Scheme α-aciclice

Noțiunea de aciclicitate, utilizată până aici, nu e altceva decât noțiunea de α-aciclicitate.

Definiția 6.7. Fie H = (N, E) un hipegraf şi n_1 şi n_2 sunt două noduri din N. Vom numi cale din n_1 în n_2 (în hipergraful H) o consecutivitate de $k\geq 1$ muchii $(E_1, ..., E_k)$, unde

(i) $n_1 \in E_1$;
(ii) $n_2 \in E_k$;
(iii) $E_i \cap E_{i+1} \neq \emptyset$ dacă $1\leq i<k$.

Vom spune, de asemenea, că consecutivitatea $(E_1, ..., E_k)$ este calea din E_1 în E_k.

Definiția 6.8. Două noduri ale hipergrafului H sunt *conexe*, dacă există o cale din unul în altul. O mulțime de noduri sau muchii sunt conexe, dacă orice două noduri sau muchii sunt conexe. *Componenta de conexiune* este mulțimea maximală de muchii conexe.

Exemplul 6.11. Considerăm hipergraful H din fig.6.19. Muchiile ABC, BCD, DE formează o cale din A în E şi din ABC în DE, de aceea A şi E, precum şi ABC şi DE sunt conexe. Mulțimile {ABC, BCD, DE} şi {IJ, JKL, IKL} sunt componente de conexiune.

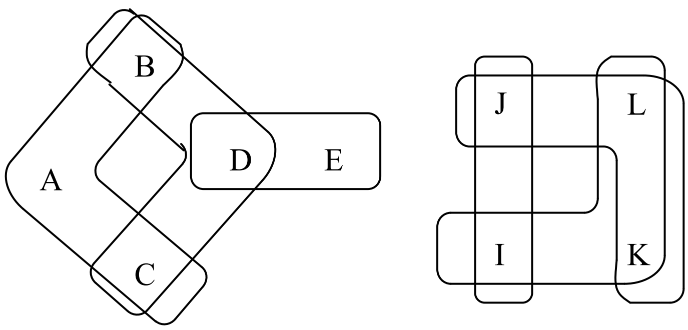

Fig.6.19. Un hipergraf cu două componente de conexiune

Noi vom examina hipergrafurile constituite dintr-o singură componentă de conexiune. Toate construcțiile şi rezultatele ulterioare se generalizează şi asupra hipergrafurilor cu mai multe componente.

Definiția 6.9. Fie hipergraful H=(N, E). Hipergraful redus (N, E^1) se obține, eliminând din E toate muchiile ce se conțin în alte muchii. Hipergraful se numeşte *redus,* dacă el este egal cu reducerea lui, adică nu poate fi eliminată nici o muchie.

Definiția 6.10. Fie E^1 o mulțime de muchii şi fie N^1 o mulțime de noduri ce apar în una sau mai multe muchii din E^1. Vom spune că E^1 este *închisă,* dacă pentru orice E_1 din hipergraf există o muchie E_2 în E^1, încât $E_1 \cap N^1 \subseteq E_2$.

Exemplul 6.12. Mulțimea de muchii {G, H, I} din fig.6.20 este o mulțime închisă. Aşadar, intersecția muchiei K cu $G \cup H \cup I$ se conține în muchia H; similar, intersecția muchiei J cu $G \cup H \cup I$ se conține în G. Însă, mulțimea {L,M} nu este închisă, datorită nodurilor x şi y. Intersecția muchiei I cu $L \cup M$ nu se conține nici în L nici în M.

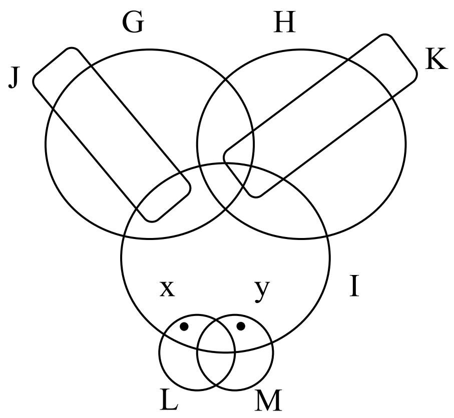

Fig.6.20

Să menționăm că orice mulțime de muchii într-un graf ordinar neorientat este închisă.

Definiția 6.11. Fie E^1 o mulțime conexă, redusă de muchii şi fie E_1 şi E_2 în E^1. Fie $Q=E_1 \cap E_2$. Vom spune că Q este o *mulțime de articulație* pentru E^1, dacă în rezultatul eliminării lui Q din orice muchie din E^1 mulțimea $\{E_i \backslash Q | E_i \in E^1\} \setminus \{\varnothing\}$ nu este conexă.

Definiția 6.12. Un hipergraf redus este *α-aciclic*, dacă orice mulțime închisă şi conexă de cel puțin trei muchii are o mulțime de articulație. Un hipergraf se spune că este α-aciclic, dacă reducerea lui este α-aciclică.

Exemplul 6.13. E simplu de verificat că hipergraful din fig.6.6 este α-aciclic. Muchiile lui sunt ABC, CDE, EFA şi ACE. O mulțime de articulație pentru toată mulțimea de muchii este $ABC \cap ACE$ = AC, fiindcă în urma eliminării lui A şi C din fiecare muchie obținem mulțimea de muchii B, DE, EF şi E ce nu sunt conexe (B nu e conexă cu celelalte). Să menționăm că mulțimea de muchii {ABC, CDE, AFA} nu are nici o mulțime de articulație. Însă, ea nu este nici închisă, deci nu este nici o contradicție cu presupunerea noastră că hipergraful din fig.6.6 este α-aciclic.

Aici putem face o legătură dintre noțiunile de dependențe joncțiune şi aciclicitate.

Definiția 6.13. Vom spune că dependența joncțiune $|\times|(R_1, ..., R_m)$ este α-aciclică, dacă α-aciclic este hipergraful format din mulțimea de muchii $\{R_1, ..., R_m\}$.

În calitate de concluzie, asupra celor spuse de la începutul acestui capitol, putem formula următoarea teoremă.

Teorema 6.7. Următoarele condiții asupra schemei bazei de date Db={R_1, ..., R_m} sunt echivalente:

(1) Schema Db este α-aciclică.
(2) Algoritmul Graham produce o mulțime vidă de muchii.
(3) Orice bază de date consistentă câte două asupra Db este consistentă.
(4) Există un program semijoncțiune de reducere completă a bazei de date cu schema Db.
(5) Schema Db are o expresie joncțiune monotonă.

6.7. Scheme β-aciclice

După cum se știe, un subhipergraf este o submulțime de muchii ale unui hipergraf. Similar, o subschemă a unei scheme a bazei de date este o submulțime de scheme relaționale din schema bazei de date. Am observat că subhipergraful unui hipergraf α-aciclic nu întotdeauna este α-aciclic. Drept exemplu pot servi hipergrafurile din fig.6.6. şi fig.6.7.

Ne interesează clasa de hipergrafuri ale căror subhipergrafuri sunt α-aciclice. Astfel de hipergrafuri se numesc β-aciclice. Importanța acestui tip de aciclicitate e greu de subestimat, fiindcă de proprietățile enumerate în teorema 6.7 se va bucura orice subschemă a bazei de date. Deci şi interpelările ce implică o parte din schemele relaționale vor fi procesate în mod eficient.

Definiția 6.14. Fie (E_1, E_2, ..., E_m, E_{m+1}) o consecutivitate de muchii ale hipergrafului H=(N, E). Presupunem că muchiile E_1, E_2, ..., E_m sunt distincte şi E_{m+1}= E_1. Consecutivitatea (E_1, E_2, ..., E_m, E_{m+1}) reprezintă un ciclu simplu, unde $1 \leq i, j \leq m$, $m \geq 3$, dacă $E_i \cap E_j \neq \emptyset$ numai în cazul când j=i+1.

În fig. 6.21, este reprezentat un ciclu simplu, ce constă din 6 muchii.

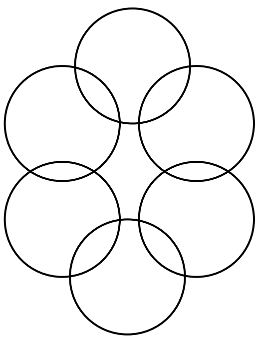

Fig.6.21. Un ciclu simplu

Definiția 6.15. Consecutivitatea de muchii (E_1, E_2,..., E_m, E_{m+1}) ale hipergrafului H=(N, E) se numeşte β-ciclu, dacă consecutivitatea (E_1^1, ..., E_m^1, E_{m+1}^1) este un ciclu simplu, unde $E_i^1 = E_i \setminus X$, $1 \leq i \leq m$ şi $X = E_1 \cap E_2 \cap ... \cap E_m$.

Evident, orice ciclu simplu este un β-ciclu.

Definiţia 6.16. Hipergraful H=(N, E) este *β-aciclic*, dacă el nu conţine un β-ciclu, în caz contrar el este β-ciclic. Schema bazei de date Db={R_1,..., R_m} este *β-aciclică (β-ciclică)*, dacă hipergraful corespunzător este β-aciclic (β-ciclic).

Definiţia 6.17. Consecutivitatea (E_1, n_1, E_2, n_2,...,E_m, n_m, E_{m+1}) de muchii şi noduri ale hipergrafului H=(N, E) se numeşte β-ciclu slab, dacă sunt satisfăcute condiţiile:

(i) n_1, n_2, ..., n_m sunt noduri distincte ale lui H;
(ii) E_1, E_2, ..., E_m sunt muchii distincte ale lui H;
(iii) $m \geq 3$;
(iv) $n_i \in E_i \cap E_{i+1}$, şi $n_i \notin E_j$ pentru orice $j \neq i$, $i+1$, $1 \leq i \leq m$.

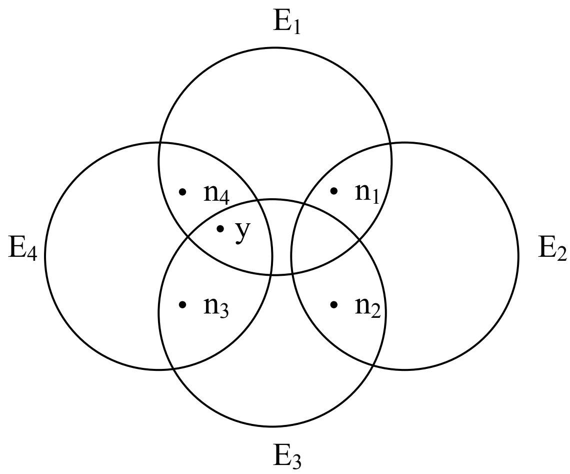

Fig.6.22. Un β-ciclu slab

Teorema 6.8. β-ciclu este un β-ciclu slab.

Demonstraţie. Fie consecutivitatea (E_1, ..., E_m, E_{m+1}) e un β-ciclu. Atunci E_1, ..., E_m, sunt distincte, $E_i \neq E_j$, $m \geq 3$ şi pentru orice două muchii vecine E_i şi E_{i+1}, $1 \leq i \leq m$, există un nod n_i în $E_i \cap E_{i+1}$, şi $n_i \notin E_j$ pentru orice $j \neq i$, $i+1$.

Afirmaţia inversă nu este corectă.

Exemplul 6.14. În fig.6.22 consecutivitatea (E_1, n_1, ..., E_4, n_4, E_1) este un β-ciclu slab, deoarece nodul $y \in E_1 \cap E_3 \cap E_4$ şi $y \notin E_2$.

Definiţia 6.18. Consecutivitatea de muchii (E_1,...,E_m, E_{m+1}) ale hipergrafului H=(N,E) se numeşte Graham-ciclu, dacă sunt satisfăcute condiţiile:

(i) E_1, ..., E_m sunt muchii distincte şi $E_1=E_{m+1}$;
(ii) $m \geq 3$;
(iii) $\Delta_i = E_i \cap E_{i+1} \neq \varnothing$, $1 \leq i \leq m$;
(iv) pentru orice $1 \leq i$, $j \leq m$, unde $i \neq j$, mulţimile Δ_i şi Δ_j sunt incomparabile, adică $\Delta_i \not\subset \Delta_j$ şi $\Delta_j \not\subset \Delta_i$.

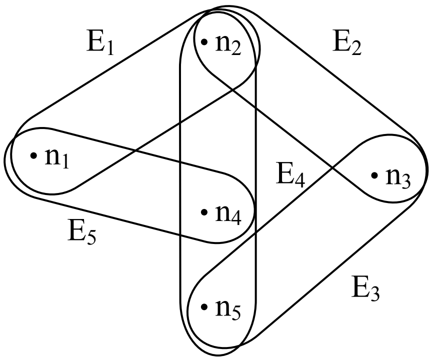

Fig.6.23. Un Graham-ciclu

Teorema 6.9. β-ciclu slab este un Graham-ciclu.

Demonstrație. Fie (E_1, n_1, ..., E_m, n_m, E_{m+1}) un β-ciclu slab. Deoarece condițiile (i), (iv) din definiția β-ciclului slab, implică condițiile (iii), (iv) din definiția Graham-ciclului, iar condițiile (ii), (iii) din definiția 6.17 coincid cu condițiile (i), (ii) din definiția 6.18, atunci consecutivitatea (E_1, ..., E_m, E_{m+1}) este Graham-ciclu.

Afirmația inversă nu este corectă.

Exemplul 6.15. În fig.6.23 consecutivitatea (E_1, E_2, E_3, E_4, E_5, E_1) este un Graham ciclu, dar nu este β-ciclu slab. Unicele β-cicluri slabe sunt (E_2, E_3, E_4, E_2) şi (E_1, E_4, E_5, E_1).

Definiția 6.19. Două muchii ale hipergrafului H=(N, E) se numesc *vecine*, dacă au noduri comune. Două muchii E_2 şi E_3 sunt *independente* în raport cu muchia E_1, dacă ele sunt vecine muchiei E_1 şi mulțimile $E_1 \cap E_2$ şi $E_1 \cap E_3$ sunt incomparabile.

Menționăm că noțiunea de independență are sens numai atunci, când este vorba de două sau mai multe muchii vecine muchiei date. În particular, muchiile care nu au deloc sau au numai un singur vecin, nu au muchii vecine independente.

Definiția 6.20. Fie hipergraful H=(N, E) şi $F \subseteq E$. Mulțimea de muchii F se *numeşte ciclu independent*, dacă ea este conexă şi fiecare muchie $E_i \in F$ are cel puțin două muchii vecine independente, E_j, $E_k \in F$.

O particularitate a ciclului independent constă în faptul că el e conceput ca o mulțime de muchii, şi nu ca o consecutivitate de muchii.

Teorema 6.10. Mulțimea de muchii ce formează un ciclu Graham este un ciclu independent.

Demonstrație. Fie consecutivitatea (E_1, E_2, ..., E_m, E_{m+1}) un ciclu Graham. Mulțimea de muchii {E_1,..., E_m} este conexă, deoarece $\Delta_j = E_j \cap E_{j+1} \neq \varnothing$, $1 \leq j \leq m$. Fiecare muchie E_k, $2 \leq k \leq m$, are în calitate de muchii vecine independente pe E_{k-1} şi E_{k+1}, fiindcă $\Delta_k \not\subset \Delta_{k+1}$ şi $\Delta_{k+1} \not\subset \Delta_k$. Pentru muchia E_1, muchiile vecine independente sunt E_2 şi E_m.

Afirmația inversă nu este corectă.

Exemplul 6.16. În fig.6.24, mulțimea de muchii $\{E_1, E_2, E_3, E_4, E_5\}$ este un ciclu independent, pe când Graham-cicluri în acest hipergraf sunt (E_1, E_2, E_3, E_1) şi (E_3, E_4, E_5, E_3).

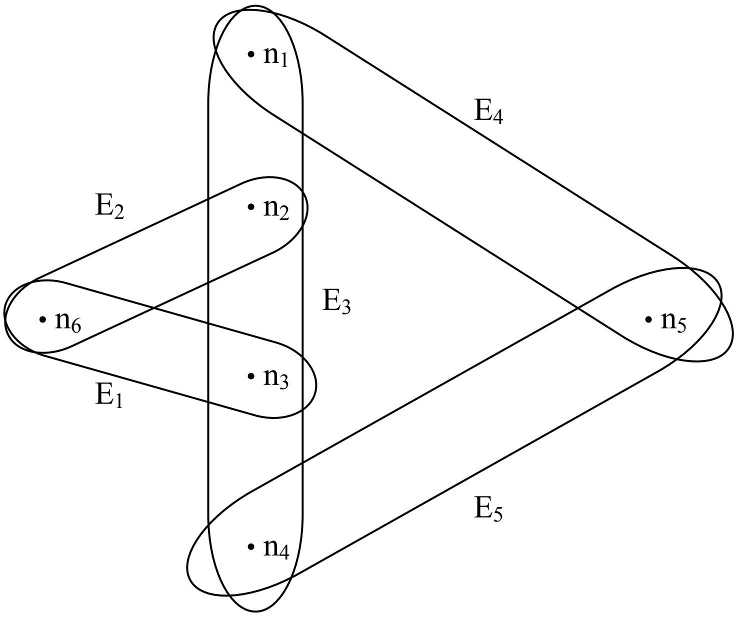

Fig.6.24. Un ciclu independent

Din teoremele 6.8, 6.9 şi 6.10 şi exemplele 6.14, 6.15, 6.16 urmează că corelația dintre noțiunile de β-ciclu, β-ciclu slab, Graham-ciclu şi ciclu independent este cea din fig. 6.25.

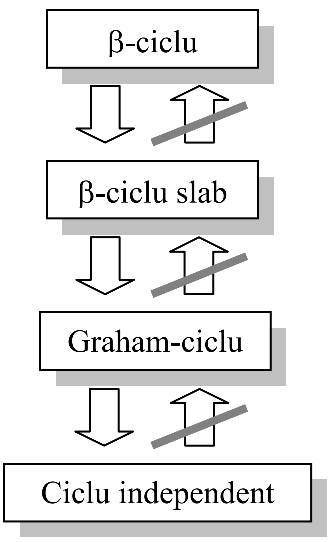

Fig.6.25. Interdependența dintre cicluri

Următoarea teoremă poate fi pusă la baza algoritmului de testare a β-aciclicităţii. Trebuie menţionat că cel mai eficient algoritm de determinare, dacă o schemă a bazei de date este β-aciclică, se bazează pe noţiunea de ciclu independent. Demonstrarea teoremei 6.11. şi descrierea algoritmului nu se aduce.

Teorema 6.11. Hipergraful H=(N, E) este β-ciclic, dacă şi numai dacă el conţine
(a) un β-ciclu, sau
(b) un β-ciclu slab, sau
(c) un Graham ciclu, sau
(d) un ciclu independent.

6.8. Scheme γ-aciclice

O schemă a bazei de date este γ-aciclică (γ-ciclică), dacă hipergraful corespunzător este γ-aciclic (γ-ciclic).

Definiţia 6.21. Consecutivitatea (E_1, n_1, E_2, n_2, ..., E_m, n_m, E_{m+1}) se numeşte *γ-ciclu* în hipergraful H=(N, E), dacă
(i) n_1, ..., n_m sunt noduri distincte în H;
(ii) E_1, ..., E_m, sunt muchii distincte în H şi $E_{m+1}=E_1$;
(iii) m>=3;
(iv) $n_i \in E_i \cap E_{i+1}$, $1 \le i \le m$ şi $n_i \notin E_j$ unde $j \neq i, i+1$ pentru $1 \le i < m$.

Să observăm că singura diferenţă dintre definiţia γ-ciclu şi β-ciclu slab este că condiţia "$1 \le i \le m$", din definiţia 6.17(iv) este substituită de condiţia "$1 \le i < m$" în definiţia 6.21(iv). Deseori e comod să ne referim la γ-ciclu, luând în consideraţie numai consecutivitatea de muchii, făcând abstracţie de noduri.

Definiţia 6.22. Un hipergraf este *γ-ciclic,* dacă conţine cel puţin un γ-ciclu.

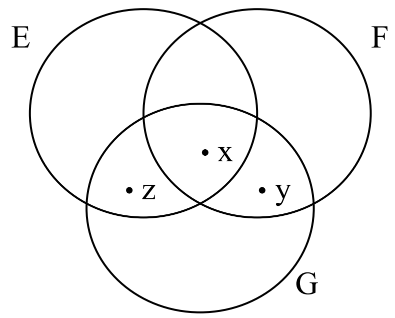

Fig.6.26. Un hipergraf γ-ciclic

Exemplul 6.17. În figura 6.26 este reprezentat un hipergraf γ-ciclic, dar β-aciclic. Într-adevăr, consecutivitatea (E, x, F, y, G, z, E) este un γ-ciclu. Pe de altă parte, E∩F∩G=y şi eliminând nodul y din fiecare muchie nu obţinem un ciclu. Deci consecutivitatea (E, x, F, y, G, z, E) nu este un β-ciclu.

Cea mai elegantă caracteristică a unui hipergraf γ-ciclic este formulată de următoarea definiție.

Definiția 6.23. Un hipergraf este γ-ciclic, dacă el conține sau un γ-ciclu de lungimea 3, sau un ciclu simplu.

Exemplul 6.18. Un ciclu simplu şi γ-ciclu de lungimea 3 sunt prezentate în fig.6.21 şi 6.26 respectiv. Să menționăm, apelând la hipergraful din fig.2.26, că într-un γ-ciclu de lungimea 3 există cel puțin un nod în intersecția $E \cap F \cap G$, cel puțin un nod în $(E \cap G) \setminus F$ şi cel puțin un nod în $(F \cap G) \setminus E$. Alte intersecții cu implicarea muchiilor E, F, G nu pot fi. Deci, dacă există un γ-ciclu de lungimea 3, atunci sau are forma ca în fig.6.26 sau ca în fig.6.21, numai cu trei muchii.

Definiția 6.24. Un hipergraf este γ-ciclic, dacă el conține o pereche E, F de muchii nondisjuncte incomparabile şi în hipergraful ce se obține din excluderea intersecției $E \cap F$, din orice muchie restul muchiei E este conexă cu restul muchiei F.

Să arătăm că aceste trei definiții ale hipergrafurilor γ-ciclice sunt echivalente.

Teorema 6.12. Definițiile 6.22, 6.23 şi 6.24 sunt echivalente.

Demonstrație. Vom arăta că (6.22)$\Rightarrow$(6.23) $\Rightarrow$(6.24) $\Rightarrow$(6.22), unde prin "(i) $\Rightarrow$(j)" subînțelegem că, dacă un hipergraf este γ-ciclic conform definiției (i), atunci este γ-ciclic conform definiției (j).

(6.22)$\Rightarrow$(6.23). Fie H este γ-ciclic, conform definiției 6.22 şi să arătăm că H e γ-ciclic şi conform definiției 6.23. Fie că $(E_1, n_1, E_2, n_2, ..., E_m, n_m, E_{m+1})$ este un γ-ciclic de lungime minimală.

Dacă m=3, atunci afirmația e demonstrată.

Presupunem că $m \geq 4$. Să arătăm că ciclul de mai sus e un ciclu simplu, adică muchiile vecine se intersectează şi muchiile nonvecine din consecutivitate nu se intersectează. Muchiile vecine se intersectează conform definiției γ-ciclu. Să demonstrăm că muchiile ce nu sunt vecine nu se intersectează.

Să arătăm că E_1 nu intersectează un nonvecin. Presupunem că se intersectează. Găsim un k ($3 \leq k < m$) cel mai mic posibil pentru care $E_1 \cap E_k \neq \emptyset$. Fie $\nu \in E_1 \cap E_k$. Atunci $(E_1, n_1, ..., E_{k-1}, n_{k-1}, E_{k-1}, E_k, \nu, E_1)$ e un γ-ciclu mai mic decât cel ipotetic. Aceasta e o contradicție.

Acum să arătăm că E_2 nu intersectează un nonvecin. Pentru aceasta presupunem că $\nu \in E_2 \cap E_k$ unde $1 \leq k \leq m$. Avem două cazuri.

Cazul 1. $\nu \in E_3$. Cunoaştem că $\nu \notin E_1$, deci E_1 nu intersectează nonvecinul E_3. Găsim r cel mai mare posibil pentru care $\nu \in E_r$. E uşor de observat că $(E_1, n_1, E_2, n_2, \nu, E_r, ..., E_m, n_m, E_1)$ este un γ-ciclu mai mic decât cel ipotetic. Deci e contradicție.

Cazul 2. $\nu \notin E_3$. Găsim r cel mai mic pentru care $\nu \in E_r$. E uşor de văzut că $(E_r, \nu, E_2, n_2, E_3, n_3, ..., E_r)$ este un γ-ciclu mai scurt decât cel ipotetic. Contradicție.

Am arătat că nici E_1 şi nici E_2 nu intersectează nonvecinii. Găsim j cel mai mic pentru care E_j intersectează un nonvecin E_k: fie $\nu \in E_j \cap E_k$. Atunci $3 \leq j$, şi $j+2 \leq k \leq m$. E uşor de văzut că $(E_1, n_1, E_2, n_2, ..., E_j, \nu, E_k, ..., E_{m+1})$ este un γ-ciclu mai scurt decât cel ipotetic. Această contradicție implică (6.22)$\Rightarrow$(6.23).

(6.23) ⇒(6.24). Fie H e γ-ciclic conform definiției 6.23. Vom arăta că H e γ-ciclic, conform definiției 6.24. Fiindcă H e γ-ciclic conform definiției 6.23, H conține sau un ciclu de lungimea 3 sau un ciclu simplu.

Presupunem că H conține un γ-ciclu de lungimea 3 şi fie acest ciclu (E_1, n_1, E_2, n_2, E_3, n_3, E_1). E uşor de verificat că H este γ-ciclic, conform definiției 6.24, (unde E şi F din fig.6.26 sunt E_1 şi E_3 respectiv).

Acum presupunem că H conține un ciclu simplu. Presupunând că E şi F sunt muchii nonvecine în ciclul simplu, vedem că H este γ-ciclic, conform definiției 6.24.

(6.24) ⇒(6.21). Fie H e γ-ciclic conform definiției (6.24). Vom arăta că H e γ-ciclic conform definiției 6.21. Luăm E şi F ca în definiția 6.24. şi fie că $Q=E\cap F$. Există o consecutivitate (E_1, ..., E_m) de muchii ce satisface condițiile:

(i) $E_1=E$,

(ii) $E_m=F$,

(iii) $(E_i \cap E_{i+1}) \setminus Q \neq \emptyset$, unde $1\leq i\leq m-1$.

Presupunem că muchiile sunt selectate în aşa fel că (i)-(iii) sunt satisfăcute pentru cel mai mic m. Dacă m=2, atunci conform (ii) $E_2=F$. Atunci $E_1\cap E_2=Q$, ce contrazice condiției (iii) pentru i=1. Prin urmare $m\geq 3$. Conform (iii), pentru orice $1\leq i\leq m-1$ în $(E_I \cap E_{i+1}) \setminus Q$ se găseşte câte un nod. Punem E_{i+1} egal E ($=E_1$) şi $n_m\in E \cap F$ (din presupunerea că $E \cap F\neq\emptyset$). Să arătăm că consecutivitatea (E_1, n_1, E_2, n_2, ..., E_m, n_m, E_1) este un γ-ciclu. Nodul n_1 nu se găseşte nici într-o muchie E_3, ..., E_{m-1}, conform minimalității lui m (deci, dacă $n_1\in E_i$, unde $3\leq i\leq m-1$, atunci consecutivitatea E_1, E_i, $E_{i+1,}$ ..., Em, trebuie folosită în locul (E_1, E_2,..., E_m). Mai departe, $n_1\notin E_m=F$. Deci $n_1\in E=E_1$; dar $n_1 \neq Q = E \cap F$. Deci n_1 se găseşte în E_1 şi E_2, şi nu în E_j, $j\neq 1,2$. Similar, n_1 este în E_i şi E_{i+1} dar nu în alt E_j, pentru $1\leq i\leq m-1$. În particular toate nodurile n_1, ..., n_{m-1} sunt distincte. În afară de aceasta n_m este distinct de toate nodurile n_1, ..., n_{m-1}, fiindcă $n_m\in Q$ şi $n_i\notin Q$, pentru $1\leq i\leq m$. Prin urmare, toate nodurile n_1 ,..., n_m sunt distincte. Din condiția că m e minimal urmează că toate muchiile E_1, ..., E_m sunt distincte. Aceasta e suficient pentru a spune că (E_1, n_1, E_2, n_2, ..., E_m, n_m, E_{m+1}) este un γ-ciclu. Deci H e γ-ciclic, conform definiției 6.22.

6.9. Proprietăți ale schemelor γ-aciclice

De noțiunea de γ-aciclicitate sunt legate o serie de proprietăți ale schemelor, proprietăți ce sunt echivalente acestei noțiuni. Vom examina doar trei proprietăți. Pentru simplitate, vom considera numai schemele bazelor de date hipergrafurile cărora constau dintr-o singură componentă de conexiune.

(1) *Orice expresie conexă de joncțiune asupra schemei bazei de date este monotonă.* Fie o schemă conexă Db. Echivalența acestei proprietăți cu γ-aciclicitate prezintă interes prin analogie cu teorema 6.6, care ne spune că schema bazei de date Db este α-aciclică atunci şi numai atunci când există o expresie joncțiune monotonă asupra Db. Deci echivalența se formulează în felul următor. Schema bazei de date Db este γ-aciclică, atunci şi numai atunci, dacă orice expresie joncțiune asupra Db este monotonă (cuvântul "conexă" a fost omis, fiindcă numai o joncțiune monotonă poate fi conexă).

Să observăm diferența dintre aceste două afirmații: în cazul α-aciclicității *există* o asemenea expresie joncțiune monotonă, pe când în cazul γ-aciclicității *orice* expresie joncțiune este monotonă.
Proprietatea de față garantează o mare libertate în luarea joncțiunilor. Aşadar, fie db este o bază de date asupra unei scheme ce se supune proprietății (1) şi este consistentă câte două. Presupunem că utilizatorul doreşte să facă joncțiunea a unei submulțimi de relații din baza de date db. Conform proprietății (1) el poate joncționa relațiile oricum doreşte fără a "dăuna" şi este sigur că a acționat în mod eficient. Prin "fără a dăuna", subînțelegem că nici o joncțiune nu va antrena relații cu scheme disjuncte, adică nu va calcula produsul cartezian. Prin "mod eficient", subînțelegem că nici o joncțiune intermediară nu va avea mai multe tupluri decât joncțiunea finală.

(2) *Fie schema bazei de date Db = $\{R_1, ..., R_m\}$. Dependența joncțiune $|\times|(R_1, ..., R_m)$ presupune că orice submulțime conexă din Db posedă proprietatea joncțiunii fără pierderi.* Adică orice dependență joncțiune inclusă $|\times|(S_1, ..., S_k)$, unde $\{S_1, ..., S_k\} \subseteq \{R_1, ..., R_m\}$, are proprietatea joncțiunii fără pierderi.

Să menționăm că această proprietate nu e caracteristică schemelor α-aciclice, fiindcă nu orice hipergraf α-aciclic este γ-aciclic.

Exemplul 6.19. Hipergraful din fig.6.2 este α-aciclic şi γ-ciclic.

(3) *În orice bază de date consistentă există o singură asociere între atributele unei mulțimi de atribute.*
Fie db=$\{r_1, ..., r_m\}$ o bază consistentă cu schema DB=$\{R_1, ..., R_m\}$. Prin asocierea atributelor din X, unde $X \subseteq R_1 \cup, ..., \cup R_m$, subînțelegem o relație $\pi_X(r_{i1}|\times|...|\times|r_{ik})$, unde $\{r_{i1},...,r_{ik}\} \subseteq$db, $X \subseteq R_{i1} \cup,...,\cup R_{ik}$ şi $\{R_{i1}, ..., R_{ik}\}$ este conexă. Adică careva din relațiile bazei de date db sunt joncționate (unde nici o joncțiune nu formează produsul cartezian) şi apoi rezultatul proiectat pe mulțimea de atribute X. Proprietatea (3) ne spune că relația rezultat este unică.

serv_funcț	FUNCȚ	DEPT	SAL
	Ionescu	Cs	150 lei

info_dept	DEPT	ORAŞ	MGR
	Cs	Chişinău	Popa

dom_funcț	FUNCȚ	STRADĂ	ORAŞ	COPII
	Ionescu	V.Lupu	Orhei	Sandu

Fig.6.27.

Exemplul 6.20. Fie schema bazei de date constă din trei scheme relaţionale: *serv_funcţ* cu atributele FUNCŢ (pentru "funcţionar"), DEPT (pentru "departament") şi SAL (pentru "salariu"); schema relaţională *info_dept* cu atributele DEPT, ORAŞ şi MGR (pentru "manager"); şi schema *dom_funcţ* cu atributele FUNCŢ, STRADĂ, ORAŞ, COPII. În fig.6.27 este reprezentată baza de date cu un tuplu în fiecare relaţie.

În acest exemplu sunt două asocieri {FUNCŢ, ORAŞ} distincte. Una, care presupune un tuplu <Ionescu, Chişinău> ce ne arată oraşul unde funcţionarul îşi are serviciul şi alta, cu tuplul <Ionescu, Orhei> ce indică locul de trai al funcţionarului. Să menţionăm că schema bazei de date este γ-ciclică, chiar α-ciclică.

Să renumim atributul ORAŞ din schema *info_dept* cu OR_SERV şi atributul ORAŞ din schema *dom_funcţ* cu OR_DOM (vezi fig.6.28). Acum avem o singură asociere {FUNCŢ, OR_SERV} cu un tuplu <Ionescu, Chişinău>, şi o singură asociere {FUNCŢ, OR_DOM} cu un tuplu <Ionescu, Orhei>.

Schema bazei de date din fig. 6.28 este γ-aciclică.

Dat fiind faptul că asocierea dintre atribute într-o schemă γ-aciclică e unică se simplifică esenţial forma interpelărilor. Cea mai simplă interpelare în limbajul SQL de găsire a tuturor funcţionarilor ce îşi au serviciul în oraşul Chişinău este

SELECT FUNCŢ
FROM *serv_funct*, *info_dept*
WHERE *serv_funct*. DEPT = *info_dept*.DEPT
AND *info_dept*.OR_SERV="Chişinău"

Dar, ţinând cont de proprietatea (3), interpelarea poate fi formulată astfel:

SELECT FUNCŢ
WHERE OR_SERV ="Chişinău"

Trebuie menţionat că avantajele nu constau numai într-o sintaxă mai simplă a interpelărilor, dar şi în posibilitatea SGBD-ului de a optimiza procesul de căutare a răspunsurilor cu o flexibilitate mai mare. Sistemul poate exploata faptul că oricare nu ar fi relaţiile în joncţiune, joncţiunea va fi monotonă şi deci eficientă.

Unele sisteme relaţionale folosesc un fişier special care să determine ce relaţii trebuie să participe la joncţiune pentru a răspunde unei interpelări. Dacă schema bazei de date este γ-aciclică, adică posedă proprietatea (3), nu e nevoie de un aşa fişier.

serv_funcţ	FUNCŢ	DEPT	SAL
	Ionescu	Cs	150 lei

info_dept	DEPT	OR_SERV	MGR
	Cs	Chişinău	Popa

dom_funcţ	FUNCŢ	STRADĂ	OR_DOM	COPII
	Ionescu	V.Lupu	Orhei	Sandu

Fig.6.28.

6.10. Algoritm de testare a γ-aciclicității

Următorul algoritm poate fi utilizat pentru verificarea dacă o schemă este sau nu γ-aciclică. El este similar celui de testare a schemelor α-aciclice.

Algoritmul constă în aplicarea în orice ordine a regulilor (a) – (e) până nu mai poate fi aplicată nici o regulă.

(a) Nodul izolat (ce aparține unei singure muchii) se elimină din muchie.
(b) Muchia unitară (ce constă dintr-un singur nod) se elimină.
(c) Muchia vidă se elimină.
(d) Dacă două muchii conțin aceleaşi noduri, se elimină una din muchii.
(e) Dacă două noduri aparțin aceloraşi muchii, atunci un nod din cele două se elimină din toate muchiile ce le conțin.

Bineînțeles că aplicarea are un număr finit de paşi. Dacă rezultatul final este o mulțime vidă de muchii, atunci hipergraful este γ-aciclic.

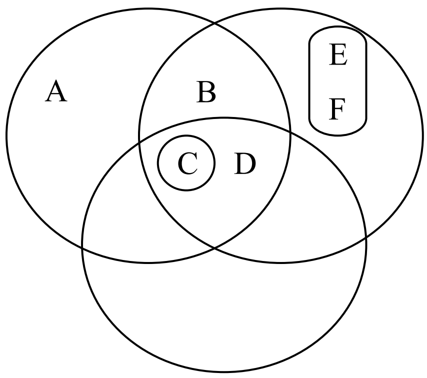

Fig.6.29.

Exemplul 6.20. Să aplicăm algoritmul asupra hipergrafului din fig.6.29. Pentru a simplifica lucrul algoritmului vom considera implicit utilizarea regulilor (d) şi (c).

La început hipergraful din fig.6.29 se prezintă:

	B	C	D	E	F
A	B	C	D		
		C			
		C	D		
				E	F

Nodul A e izolat şi muchia {C} e unitară, deci conform regulilor (a) şi (b) sunt eliminate. Au rămas muchiile:

B	C	D	E	F
B	C	D		
	C			
	C	D		
			E	F

Nodurile E şi F aparțin împreună aceloraşi muchii (BCDEF şi EF) şi conform regulii (e) nodul F se elimină din ambele muchii. Similar, nodurile C şi D aparțin aceloraşi muchii. Se elimină nodul D din cele trei muchii. Au rămas muchiile:

B	C	E
B	C	
	C	
		E

Se elimină a treia şi a patra muchie, fiindcă sunt unitare:

B	C	E
B	C	

Nodul E e izolat. După eliminarea nodului E au rămas muchiile:

B	C
B	C

Aceste două muchii sunt identice. Conform regulii (d) o muchie se elimină:

B	C

Întrucât ambele noduri sunt izolate, ele sunt eliminate. În rezultat s-a obţinut o mulţime vidă de muchii, deci hipergraful din fig.6.29 este γ-aciclic. Deci şi schema asociată lui este γ-aciclică.

6.11. Scheme Berge-aciclice

Vom considera în această secţiune unul din cele mai puternice tipuri de aciclicitate ale schemelor bazei de date şi anume Berge-aciclicitatea.

Definiţia 6.25. Fie hipergraful H=(N, E). Consecutivitatea (E_1, n_1, E_2, ..., E_m, n_m, E_{m+1}) se numeşte *Berge-ciclu*, dacă sunt satisfăcute condiţiile:
(i) $n_1, ..., n_m$ sunt noduri distincte în N;
(ii) $E_1, ..., E_m$ sunt distincte în E şi $E_1=E_{m+1}$;
(iii) $m>1$;
(iv) $n_i, n_{i+1} \in E_i$, $0<i<m+1$.

Definiţia 6.26. Hipargraful H=(N, E) este Berge-ciclic, dacă conţine cel puţin un Berge-ciclu, în caz contrar e Berge-aciclic.

Exemplul 6.21. Hipergraful din fig.6.30 este Berge-ciclic, dar γ-aciclic. Un Berge-ciclu este consecutivitatea (ABC, B, BCD, C, ABC). Consecutivitatea dată nu e γ-ciclu, fiindcă sunt implicate în ea numai două muchii.

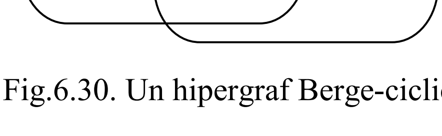

Fig.6.30. Un hipergraf Berge-ciclic

Exemplul 6.21 ne sugerează următoarea teoremă.

Teoremă 6.13. Dacă o pereche de muchii ale unui hipergraf au două sau mai multe noduri comune, atunci hipergraful este Berge-ciclic.

Demonstrație. Afirmația rezultă imediat din definiția 6.25.

O procedură de testare, dacă o schemă este Berge-aciclică, constă în aplicarea următoarelor două reguli:

(a) Nodul izolat (ce aparține unei singure muchii) este eliminat din muchie;

(b) Muchia unitară (ce constă dintr-un singur nod) sau vidă se elimină.

Algoritmul sfârşeşte, dacă nu mai poate fi aplicată nici una din reguli. Dacă algoritmul produce o mulțime vidă de muchii, atunci hipergraful inițial este Berge-aciclic, în caz contrar e Berge-ciclic.

6.12. Exerciții

6.1. Fie schema bazei de date Db = {ABC, BCE, CDE}. Să se aducă un exemplu de bază de date db = {r_1(ABC), r_2(BCE), r_3(CDE)} şi a două programe semijoncțiune de reducere completă, astfel că aplicarea unui program reduce costul calculării $r_1|\times|r_2|\times|r_3$, iar aplicarea altui program nu e eficientă. Să se considere că costul transferării unei valori atomice e 1 şi costul calculării $r_i|\times r_j$ este egal cu costul transferării proiecției relației r_j asupra schemei relației r_i, iar joncțiunea trebuie să se efectueze în punctul de păstrare a relației r_1.

6.2. Poate să se schimbe relația r în urma atribuirii de forma $r \leftarrow r|\times s$, dacă schemele relațiilor r şi s sunt disjuncte?

6.3. Să se arate că pentru schema bazei de date Db={ABC, BCD, DE} nu există o expresie de joncțiune monotonă.

6.4. Să se determine care din următoarele scheme ale bazei de date sunt α-aciclice şi care α-ciclice.

(a) Db_1 = {ABC, CDE, AIE, ACE};

(b) Db_2 = {ABC, BCD, ACD, ABD};

(c) Db_3 = {AB, BD, CD, CE, DE}.

6.5. Să se arate că schema bazei de date Db = {ABCD, CDE, DEF, EFG, GK, BCDIL, MIL, EN, NO} este β-aciclică.

6.6. Să se determine cel mai înalt graf de aciclicitate a schemelor de mai jos:

(a) Db1 = {ABCD, BCE, CDF};

(b) Db2 = {ABC, CDE, DF, EGH, B};

(c) Db3 = {ABCD, CDEG, BDEF};

(d) Db4 = {CH, ACD, CDF, BDE, DEG, BI};

(e) Db5 = {BCE, ABC, CEG, BEF};