

# TEHNICI DE EXPLORARE A DATELOR -DATA MINING -

CURS 9

## *Descoperirea regulilor de asociere booleene*

- două sarcini separate
  - găsirea mulțimilor de *articole frecvente*, care depășesc pragul minim impus pentru suport *minsup*.
    - este etapa care consumă cel mai important volum de resurse
  - pe baza acestora se vor construi regulile de asociere care au confidența mai mare decât minimul stabilit prin valoarea *minconf*.

# Algoritmul Apriori

- *D*- baza de date de tranzacții
- *t*- tuple din *D*
- *k-itemset*-o mulțime cu *k* articole
- *L<sub>k</sub>*- mulțimea de *k*-itemsets frecvente( cele cu suport mai mare decât pragul minim). Fiecare membru al acestei mulțimi are doua câmpuri: setul de articole si valoarea suportului.
- *C<sub>k</sub>*- mulțimea de *k*-itemsets candidate(potențial frecvente). Fiecare membru al acestei mulțimi are doua câmpuri: setul de articole si valoarea suportului.

$L_1 = [\text{itemseturi frecvente de dimensiune } 1];$

**k=2**

**cât timp**  $L_{k-1} \neq \Phi$  **execută**

$C_k = \text{gen\_apriori}(L_{k-1})$  //candidate noi

**pentru**  $\forall t \in D$  **execută**

$C_t = \{C_k | C_k \subset t\}$  //candidate conținute în *t*

**pentru**  $\forall c \in C_t$  **execută**

$c.\text{count} = c.\text{count} + 1$

$L_k = \{c \in C_k | c.\text{count} \geq \text{minsup}\}$

$k = k + 1$

**sfârșit**

Funcția **gen-apriori** ia ca argument  $L_{k-1}$ , mulțimea tuturor seturilor frecvente de  $(k-1)$ -articole. Ea întoarce o supramulțime a tuturor mulțimilor frecvente care conțin  $k$  articole. Funcția se execută în două faze:

- pasul de uniune care face o uniune a  $L_{k-1}$  cu  $L_{k-1}$ :

```
insert into Ck
select p.item1, .., p.itemk-1, q.itemk-1
from Lk-1 p, Lk-1 q
where p.item1=q.item1 and ... and
p.itemk-2=q.itemk-2 and p.itemk-1<q.itemk-1 ;
```

- pasul de eliminare presupune ștergerea tuturor mulțimilor de articole  $c \in C_k$  pentru care anumite submulțimi de  $(k-1)$  articole din  $c$  nu sunt în  $L_{k-1}$

**pentru**  $\forall$  mulțime de articole  $c \in C_k$  **executa**

**pentru**  $\forall$  submulțime de dimensiune  $(k-1)$   $s \in c$  **executa**

**daca**  $(s \notin L_{k-1})$  **atunci**

**șterge**  $c$  din  $C_k$  ;



# Generarea regulilor

- $R$  numărul tuturor regulilor posibil de extras dintr-o mulțime de literali distincți  $I$  cu  $|I|=n$ :  
 $|R| = 3^n - 2^{n+1} + 1$
- $n$  poate lua valori de la ordinul sutelor până la cel al sutelor de mii, caz în care, explorarea tuturor regulilor este imposibilă.
- -> regulile se extrag doar din multimea *itemseturilor frecvente*

**Intrări:** toate itemseturile frecvente cu dimensiunea cel puțin 2

**Ieșiri:** toate regulile de asociere

// Algoritm simplu de descoperire a regulilor de asociere

\*) **pentru** toate  $F_k$ ,  $k \geq 2$

genreg( $F_k, F_k$ );

//procedura genreg generează toate regulile valide  $\tilde{a} \Rightarrow (l_k - \tilde{a})$ , pentru toate  $\tilde{a} \subset a_m$

**Intrări:**  $l_k, a_k$  - două mulțimi de articole frecvente

$cmin$  - o valoare minima pentru confidență

**Ieșiri:** regulile de asociere cu maxim  $m-1$  articole în antecedent ( $m > 2$ )

**procedura** genreg( $l_k$ :k-itemset,  $a_m$ :m-itemset) **este**

$A = \{a_{m-1} \mid a_{m-1} \subset a_m\};$

**pentru** toate  $a_{m-1} \in A$  **execută**

$conf = \sup(l_k) / \sup(a_{m-1})$

**dacă** ( $conf \geq cmin$ ) **atunci execută**

**scrie** regula  $a_{m-1} \Rightarrow (l_k - a_{m-1})$  cu  $confidentia = conf$  si  $suportul = \sup(l_k)$

**dacă** ( $m-1 > 1$ ) **atunci**

genreg( $l_k, a_{m-1}$ );

//generează regulile cu submulțimi ale mulțimii  $a_{m-1}$  în antecedent

**sfarsit**

**sfârșit**

**sfârșit procedura**

# Exemplu

Baza de date D		Candidate 1-itemset		Frecvente 1-itemset	
TID	Articole	Itemset	Suport	Itemset	Suport
100	A C D	{A}	2	{A}	2
200	B C E	{B}	3	{B}	3
300	A B C E	{C}	3	{C}	3
400	B E	{D}	1	{E}	3
		{E}	3		

TID	Items			
100	A	C	D	
200	B	C	E	
300	A	B	C	E
400	B	E		

Candidate 2-itemset		Candidate 2-itemset		Frecvente 2-itemset	
Itemset		Itemset	Suport	Itemset	Suport
{A, B}		{A, B}	1	{A, C}	2
{A, C}		{A, C}	2	{B, C}	2
{A, E}		{A, E}	1	{B, E}	3
{B, C}		{B, C}	2	{C, E}	2
{B, E}		{B, E}	3		
{C, E}		{C, E}	2		

Candidate 3-itemset		Candidate 3-itemset		Frecvent 3-itemset	
Itemset		Itemset	Suport	Itemset	Support
{B, C, E}		{B, C, E}	2	{B, C, E}	2

B si C $\Rightarrow$ E,	cu suportul = 50%	si confidenta = 100%
B si E $\Rightarrow$ C,	cu suportul = 50%	si confidenta = 66.7%
C si E $\Rightarrow$ B,	cu suportul = 50%	si confidenta = 100%
B $\Rightarrow$ C and E,	cu suportul = 50%	si confidenta = 66.7%
C $\Rightarrow$ B and E,	cu suportul = 50%	si confidenta = 66.7%
E $\Rightarrow$ B and C,	cu suportul = 50%	si confidenta = 66.7%

minisup = 50%

miniconf = 60%

## Optimizări ale algoritmului Apriori

- algoritmul Apriori necesită un număr de scanări ale bazei de date egal cu dimensiunea celei mai mari mulțimi de articole frecvente.
- doi factori determinanți ai performanțelor sunt:
  - numărul trecerilor prin baza de date
  - eficiența acestor treceri
- au fost propuse variante ale algoritmului Apriori, care au drept caracteristică, reducerea numărului de scanări ale bazei de date în scopul numărării mulțimilor de articole.

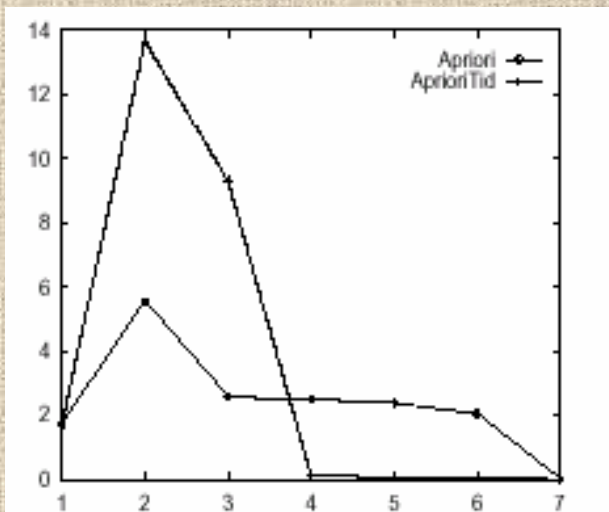


# Optimizări ale algoritmului Apriori

## AprioriTID și AprioriHibrid

### AprioriTID

- Utilizează funcția *AprioriGen* pentru a determina mulțimea de articole candidat.
- Caracteristici:
  - baza de date  $D$  este utilizată pentru determinarea suportului numai în primul pas.
  - în etapele următoare sunt utilizate mulțimile  $C'_k$ .
- Fiecare element al mulțimii  $C'_k$  este de forma  $\langle \text{TID}, \{X_k\} \rangle$ , unde fiecare  $X_k$  este o mulțime potențial frecventă de articole de dimensiune  $k$ , ce se găsește în tranzacția cu identificatorul TID.



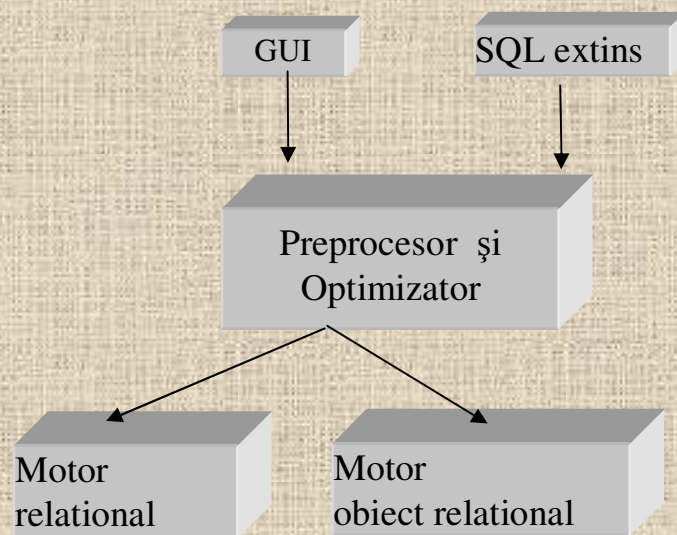
### AprioriHibrid- combină algoritmi Apriori și AprioriTID

- Comutarea se face în momentul în care mulțimea candidatelor generate scade astfel încât începe să poată fi păstrată în memorie
- Implica costuri suplimentare
- Reduce timpii de execuție



# Utilizarea SQL pentru determinarea regulilor de asociere

- **Avantaje:**
  - se pot utiliza facilitățile de indexare și cele de procesare a interogărilor proprii sistemului
  - se poate exploata posibilitatea de execuție paralelă a instrucțiunilor SQL, de exemplu, într-un mediu SMP



Model de explorarea datelor într-un SGBD prin folosirea SQL

## Utilizarea SQL pentru determinarea regulilor de asociere

- Algoritmul Apriori presupune execuția iterativă a două faze.
  - din itemset-urile care aparțin mulțimii  $F_{k-1}$  se va genera  $C_k$  (mulțimea candidatelor de dimensiune  $k$ ). Aceasta se realizează prin următoarea operație de joncțiune:

```
insert into  $C_k$ 
select  $M_1.item_1, \dots, M_1.item_{k-1}, M_2.item_{k-1}$ 
from  $F_{k-1} M_1, F_{k-1} M_2$ 
where  $M_1.item_1 = M_2.item_1$  and ... and
 $M_1.item_{k-2} = M_2.item_{k-2}$  and
 $M_1.item_{k-1} < M_2.item_{k-1}$  ;
```

- din  $C_k$  sunt înlăturate acele itemset-uri în care există cel puțin un subset de dimensiune  $k-1$  care nu se găsește în  $F_{k-1}$ .
- Utilizând SQL se pot executa cele două faze într-o singură comandă - se face joncțiunea mulțimii  $L_{k-1}$  cu ea însăși de  $k$  ori și se impun condiții care să elimine din rezultatul final acele itemseturi care conțin submulțimi de dimensiune  $k-1$  care nu sunt frecvente.



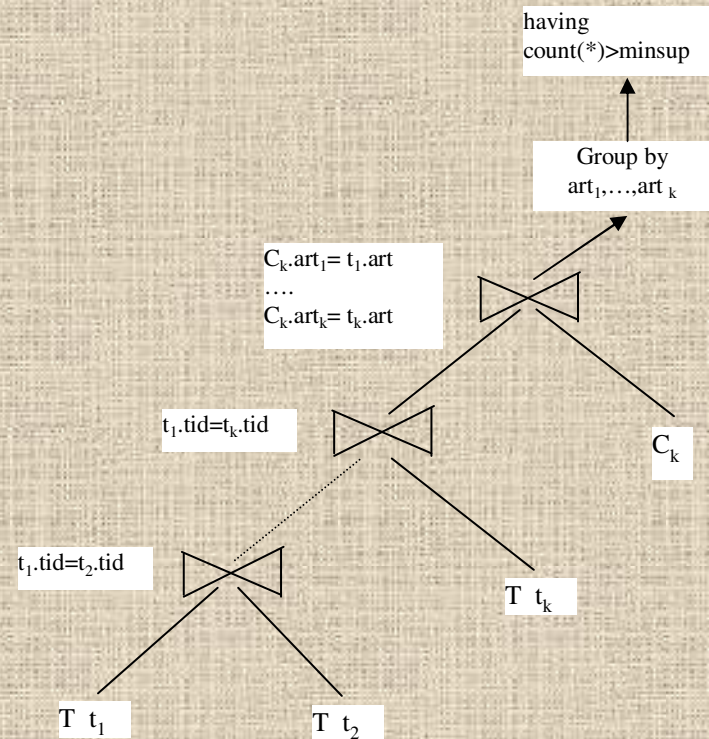


Pentru *calculul suporturilor* :

•în fiecare pas se face joncțiunea itemset-urilor din  $C_k$ , de  $k$  ori cu tabelul de tranzacții (T) urmată de o grupare după item-uri,

Comanda SQL care rezolvă acest lucru este următoarea:

```
insert into Fk
select art1,...artk,count(*)
from Ck,T t1,...T tk
where t1.art = Ck.art1 and
. . .
tk.art = Ck.artk and
t1.tid = t2.tid and
. . .
tk-1.tid = tk.tid
group by art1,art2 . ..artk
having count(*)>minsup
```



Arborele de analiză al interogării care permite determinarea itemseturilor frecvente

Pentru a genera regulile care au confidența cel puțin egală cu pragul *minconf*, se vor găsi mai întâi toate submulțimile nevide ale fiecărui itemset frecvent. Apoi, pentru fiecare submulțime  $m$  se va calcula confidența regulii  $m \rightarrow (k-m)$ , și în cazul în care aceasta este cel puțin *minconf* se reține regula.

**Intrări:** toate itemseturile frecvente cu dimensiunea cel puțin 2

**Ieșiri:** toate regulile de asociere

\* ) **pentru** toate  $F_k$ ,  $k \geq 2$   
       genreg( $F_k, F_k$ );

**//procedura genreg generează toate regulile valide  $\tilde{a} \Rightarrow (l_k - \tilde{a})$ , pentru toate  $\tilde{a} \subset a_m$**

**Intrări:**  $l_k, a_k$ - două mulțimi de articole frecvente

*cmin* - o valoare minima pentru confidență

**Ieșiri:** regulile de asociere cu maxim  $m-1$  articole în antecedent ( $m > 2$ )

**procedura** genreg( $l_k$ : itemset dedimensiunea  $k$ ,  $a_m$ : itemset dedimensiunea  $m$ ) **este**

$A = \{a_{m-1} \mid a_{m-1} \subset a_m\};$

**pentru** toate  $a_{m-1} \in A$  **execută**

conf = sup( $l_k$ ) / sup( $a_{m-1}$ )

**dacă** (conf  $\geq$  cmin) **atunci execută**

**scrie** regula  $a_{m-1} \Rightarrow (l_k - a_{m-1})$  cu confidența=conf și suportul=sup( $l_k$ )

**dacă** ( $m-1 > 1$ ) **atunci**

          genreg( $l_k, a_{m-1}$ );

**//generează regulile cu submulțimi ale mulțimii  $a_{m-1}$  în antecedent**

**sfârșit**

**sfârșit**

**sfârșit procedura**

În faza de numărare a suportului itemset-urile frecvente de o anumită dimensiune sunt memorate într-un tabel. De exemplu itemseturile de dimensiune 1 sunt memorate în tabelul  $F_1$ , cele de dimensiune 2 în tabelul  $F_2$  ș.a.m.d.

În vederea generării regulilor vor fi fuzionate toate aceste tabele într-unul singur  $F$ . Schema acestui tabel va conține  $k+2$  attribute și anume ( $art_1, art_2, \dots, art_k, suport, dimens$ ) unde  $k$  este lungimea celui mai mare itemset frecvent, *suport* este valoarea calculată a suportului iar *dimens* este lungimea fiecărui itemset.

Pentru obținerea regulilor de asociere, pornind de la  $F$  se creează un tabel intermediar  $RT$  cu următoarea structură ( $art_1, art_2, \dots, art_k, suport, dimens, poz$ ), în care atributul  $poz$  indică poziția simbolului „ $\rightarrow$ ” în regulă.

Pentru a calcula confidența regulii ca un raport între suportul regulii și suportul antecedentului, precum și pentru a determina suportul antecedentului se va face o joncțiune între tabelele  $RT$  și  $F$ , după cum urmează:



```

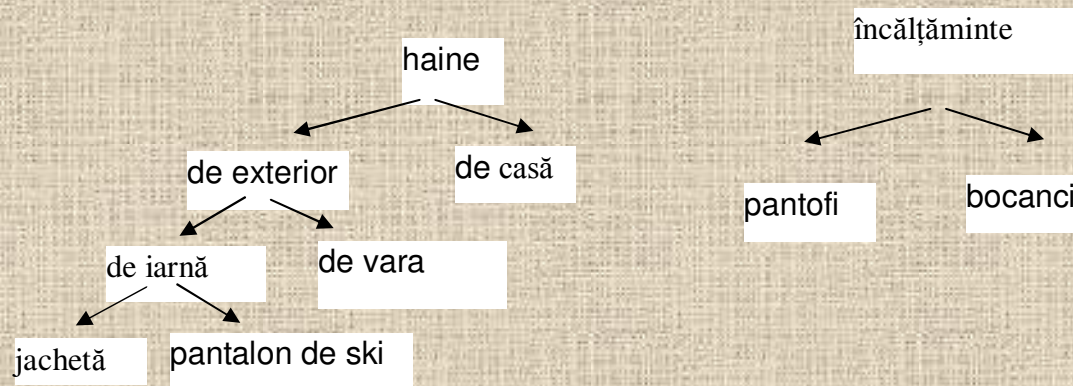
insert into R
  Select RT.art1,...RT.artk, RT.suport, RT.dimens,RT.poz,
    RT.suport/F.suport
    from RT, F
      where (RT.art1 = F.art1 or RT.poz>1)
          . . .
          AND (RT.artk = F.artk or RT.poz>k)
          AND RT.poz = F.dimens
          AND RT.suport/F.suport > minconf;

```

# Alti algoritmi de explorare a regulilor de asociere booleene

- **DHP (Direct Hashing and Pruning)**: plasarea contoarelor pentru suport intr-o structura hash si eliminarea tranzactiilor pe baza informatiilor din aceasta structura
- **FP-growth**: determinarea itemseturilor frecvente fara generarea de candidate
- **DIC (Dynamic Itemset Counting)**: adaugarea de noi itemseturi candidate in diferite momente ale scanarii
- **Partition**: Partitionarea datelor pentru a gasi itemseturile candidate

# Reguli de asociere extinse - R.A. generalizate



- Presupun existența unei taxonomii a articolelor - o ierarhie "isa" de articole care permit descoperirea de asociații între orice nivele ale ierarhiei.
- interesantă generarea regulilor care pun în relație diferitele nivele ale taxonomiei

## •utile din următoarele motive:

- regulile stabilite la nivele joase pot să nu depășească pragul minim pentru suport. In multe domenii de aplicare numărul articolelor poate fi foarte mare, astfel încât, dacă ele sunt organizate în categorii pot fi detectate reguli interesante care altfel ar fi pierdute.
- Taxonomiile pot fi utilizate pentru a reduce regulile *neinteresante* sau *redundante*.

## Rezolvarea problemei se descompune în trei pași:

- găsirea tuturor mulțimilor de articole frecvente
  - utilizarea acestor mulțimi pentru a genera mulțimea regulilor dorite
  - eliminarea tuturor regulilor neinteresante din mulțimea de reguli generată
- Metoda de bază constă în aplicarea algoritmului Apriori asupra mulțimii tranzacțiilor extinse  $T'$  obținută prin adăugarea tuturor strămoșilor fiecărui articol din  $T$  în  $T$

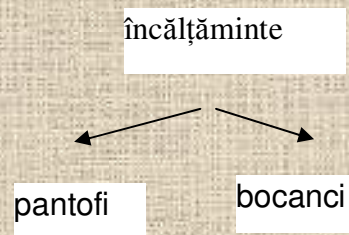
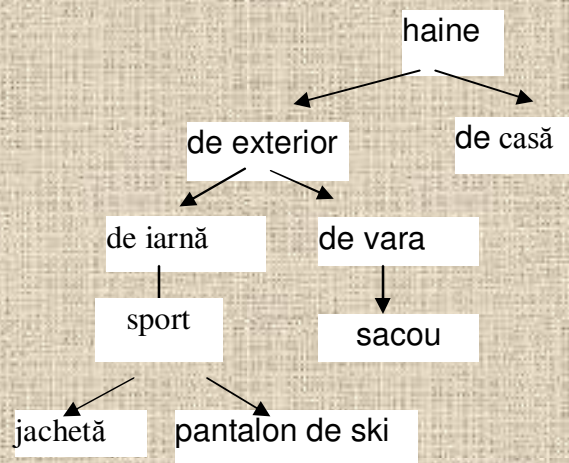


# Formalizarea problemei

- Fie  $I = \{I_1, \dots, I_m\}$  o mulțime de literali, numite articole, și fie  $T$  un graf aciclic direcționat (DAG) pe acești literali.
  - Un arc în  $T$  reprezintă o legătură *isa*, iar  $T$  reprezintă o mulțime de taxonomii
  - Dacă  $x'$  și  $x$  sunt unite printr-un arc, atunci  $x'$  va fi numit părinte(ancestor, stramos), iar  $x$  va fi numit copilul (descendent) lui  $x'$ . Altfel spus,  $x'$  reprezintă o generalizare a lui  $x$ .
- Fie  $D$  un set de tranzacții, unde fiecare tranzacție  $T$  este o mulțime de articole astfel încât  $T \subseteq I$ . O tranzacție  $T$  suportă un articol  $x' \in I$  dacă  $x' \in T$ , sau  $x'$  este un ancestor al unui articol  $x \in T$ . O tranzacție  $T$  suportă o mulțime de articole  $X \subseteq I$  dacă  $T$  suportă orice articol din  $X$ .
- O regulă de asociere generalizată este o implicație de forma  $X \rightarrow Y$ , unde  $X \subset I$ ,  $Y \subset I$  și  $X \cap Y = \emptyset$ , și nici un articol din  $Y$  nu este ancestor al vreunui articol din  $X$ . O regulă de forma  $x \rightarrow \text{ancestor}(x)$  are confidența 100%, este **trivială și redundantă**.
- Regulile se numesc generalizate deoarece atât  $X$  cât și  $Y$  pot conține articole aflate pe niveluri diferite ale taxonomiei  $\mathcal{I}$ .

# Exemplu

- Se consideră mulțimea  $I = \{\text{încălțăminte, pantofi, bocanci, haine, haine de exterior, haine de casa, haine de iarna, haine de vara, sport, jachetă, pantalon de schi, sacou}\}$  și  $\mathcal{T}$  o taxonomie a acestor articole
- se consideră următoarea baza de date tranzacțională.



Identificator tranzacție	Mulțime de articole
1	sacou
2	jachetă, bocanci
3	pantaloni de ski, bocanci
4	pantofi
5	pantofi
6	jachetă

Mulțime de articole	Suport
{jachetă}	2 (33%)
{de iarnă}	3 (50%)
{sport}	3 (50%)
{haine}	4 (66%)
{pantofi}	2 (33%)
{bocanci}	2 (33%)
{încălțăminte}	4 (66%)
{de iarnă, bocanci}	2 (33%)
{sport, bocanci}	2 (33%)
{haine, bocanci}	2 (33%)
{de iarnă, încălțăminte}	2 (33%)
{sport, încălțăminte}	2 (33%)
{haine, încălțăminte}	2 (33%)

Mulțimi de articole frecvente

Reguli	Suport	Confidență
de iarnă → bocanci	33%	66%
sport → bocanci	33%	66%
de iarnă → încălțăminte	33%	66%
sport → încălțăminte	33%	66%
bocanci → de iarnă	33%	100%
bocanci → sport	33%	100%
bocanci → haine	33%	100%

Reguli de asociere

minsup=30% (minim două tranzacții)  
minconf=60%

Se observă că reguli de genul *pantalon de ski* → *bocanci* sau *jachetă* → *bocanci* nu îndeplinesc condiția de suport minim, în timp ce regulile *de iarnă* → *bocanci* și *sport* → *bocanci* o îndeplinesc.

# Alte tipuri de reguli de asociere extinse

- Reguli de asociere ponderate
- Reguli de asociere cantitative
- Reguli de asociere in baze de date foarte mari
- Reguli de asociere in baze de date dinamice



Simplificarea tabelelor și a grafului în scopul focalizării asupra celor mai interesante reguli

