# Apunts del Taller de Nous Usos de la Informàtica

Jordi Vitrià

Universitat de Barcelona

10 de setembre de 2019

UNIVERSITAT DE BARCELONA

# Lliçó: El problema dels ítems freqüents

# El model de la bossa de la compra

- Donat un gran conjunt d'ítems (per exemple, els productes que es venen a un supermercat)
- Donat un gran conjunt de clients d'un dia determinat, cada un amb la seva bossa de la compra
- **Podem trobar connexions interessants entre els ítems a partir d'analitzar el contingut de les bosses?**
- Aquest tipus de relació de "molts-a-molts"es pot aplicar a molts tipus de problemes, és molt general!

# Conjunts d'ítems freqüents

- La pregunta més simple que ens podem fer és: Quins són els subconjunts d'ítems que apareixen de manera més freqüent a les bosses?
- Definim el *suport* d'un (sub)conjunt d'ítems $I$ com el nombre de bosses que contenen tot el conjunt $I$.
- Donat una valor $s$, anomenem *conjunt d'ítems freqüents* a tots els conjunts d'ítems que tenen un suport de com a mínim $s$.

# Exemple: Conjunts d'ítems freqüents

- Ítems = { m, c, p, b, j }.
- Suport: $s = 3$ bosses.
- Les bosses:

$$B_1 = \{ \text{m, c, b} \} \quad B_2 = \{ \text{m, p, j} \}$$
$$B_3 = \{ \text{m, b} \} \quad B_4 = \{ \text{c, j} \}$$
$$B_5 = \{ \text{m, p, b} \} \quad B_6 = \{ \text{m, c, b, j} \}$$
$$B_7 = \{ \text{c, b, j} \} \quad B_8 = \{ \text{b, c} \}$$

- El conjunts d'ítems freqüents són: { m }, { c },{ b },{ j },{ m, b },{ b, c } { c, j }, i no n'hi ha més.

# Aplicacions

- Si els ítems són els productes d'un supermercat i les bosses són el que un client compra un dia determinat, una de les connexions que podríem trobar és, per exemple, que molta gent compra cervesa i bolquers.
- Aquesta informació pot servir per collocar-los junts i fer una oferta sobre els bolquers i al mateix temps pujar el preu de la cervesa per compensar (els humans sóm així!).
- Aquesta estratègia només és òtil si hi ha molta gent que compra cervesa i bolquers (per exemple, un 1%).

# Aplicacions

- Si els ítems són documents, les bosses són frases contingudes als documents, i definim que un item/document està a una bossa/frase si la frase està al document, els ítems que apareixen junts massa sovint poden indicar *plagiarisme*.
- Si les bosses són pàgines web i els ítems paraules, la coocurrència de paraules *rares* (en el sentit de poc probables, com *Brad* i *Angelina*) pot indicar relacions interessants.

# Aplicacions

- Si les bosses són pacients que han sofert efectes no desitjats després d'un tractament i els ítems són medicaments podem detectar interaccions perilloses entre medicaments.

# Us: Generació de regles d'associació

- Aquesta anàlisi pot generar regles del tipus *Si-Llavors* sobre els continguts de les bosses.
- $\{i_1, i_2, i_3, \ldots, i_k\} \rightarrow j$ significa que si la bossa conté els ítems $i_1, i_2, i_3, \ldots, i_k$ llavors és molt probable que contingui $j$.
- La **confiança** d'aquesta *regla d'associació* Es la probabilitat condicional de $j$ donats $i_1, i_2, i_3, \ldots, i_k$.

$$C = \frac{\text{Suport } \{i_1, i_2, i_3, \ldots, i_k\} \cup \{j\}}{\text{Suport } \{i_1, i_2, i_3, \ldots, i_k\}}$$

# Regles d'associació interessants

- No totes les regles d'associació són interessants. Per exemple, la regla $X \rightarrow \textit{llet}$ pot tenir una confiança alta només perquè molta gent compra llet, independentment de $X$.
- L'**interès** es defineix com la diferència entre la seva *confiança* i la fracció de bosses (sobre el total) que contenen $j$.

$$I = C - \frac{\text{Suport}\{j\}}{n}$$

- Les regles $\{i_1, i_2, i_3, \ldots, i_k\} \rightarrow j$ **interessants són les que tenen un valor molt positiu o negatiu d'interès**.

# Regles d'associació interessants

- Si $\{i_1, i_2, i_3, \ldots, i_k\}$ no té influència sobre $j$ tindrem tantes bosses amb $\{i_1, i_2, i_3, \ldots, i_k, j\}$ com $\{i_1, i_2, i_3, \ldots, i_k\}$.
- Per tant, *bolquers* $\rightarrow$ *cervesa* té un interès alt, *coke* $\rightarrow$ *pepsi* i *pepsi* $\rightarrow$ *coke* tenen un interès molt baix (negatiu) i $X \rightarrow$ *llet* té un interès 0.

# Exemple: Confiança i interès

$$
\begin{array}{ll}
B_1 = \{ \text{m, c, b} \} & B_2 = \{ \text{m, p, j} \} \\
B_3 = \{ \text{m, b} \} & B_4 = \{ \text{c, j} \} \\
B_5 = \{ \text{m, p, b} \} & B_6 = \{ \text{m, c, b, j} \} \\
B_7 = \{ \text{c, b, j} \} & B_8 = \{ \text{b, c} \}
\end{array}
$$

- La regla d'associació $\{m, b\} \rightarrow c$ té una confiança 0.5 (hi ha dos casos sobre els quatre possibles) i un interès -0.125 ($= 0.5 - 5/8$).

# Cerca de regles d'associació

- El problema és *trobar totes les regles d'associació amb suport* $\geq s$ *i confiança* $\geq c$. (El suport de la regla és el suport del conjunt d'ítems de l'esquerra)
- La part més dura és trobar els conjunts d'ítems més frequents:
  - Si $\{i_1, i_2, i_3, \ldots, i_k\} \rightarrow j$ té suport (1%) i confiança (50%) elevats llavors tant $\{i_1, i_2, i_3, \ldots, i_k\}$ com $\{i_1, i_2, i_3, \ldots, i_k, j\}$ seran freqüents.
  - Hem de trobar tots els conjunts d'ítems amb suport alt i després totes les regles d'associació amb suport i interès suficients.
  - Si en conjunt d'ítems freqüents tenim $j$ elements, podem generar $j$ possibles regles.

# Model Computacional

- Típicament les dades les tindrem en un fitxer en un disc, emmagatzemades *per bosses* i hem d'expandir les bosses en parells, triples, etc. a mesura que anem llegint les bosses.
- La implementació ingènua és usar un conjunt de $k$ iteradors niuats per generar tots els conjunts de mida $k$.

# Model Computacional

- El cost més important a l'hora de processar aquestes dades és el nombre de lectures/escriptures al disc.
- A la pràctica, els algorismes que generen regles d'associació llegeixen les dades en *passades*, llegint totes les bosses una darrera l'altra.
- Per tant, mesurarem el cost de l'algorisme pel nombre de passades que fa sobre el conjunt total de les dades.

# Model Computacional

- Per aquests algorismes, l's de la memòria principal és el recurs més crític.
  - A mesura que llegim bosses, hem de contar alguna cosa (com per exemple els cops que apareix una determinada parella d'ítems).
  - El nombre de coses diferents que podem contar està limitat per la memòria principal.

# Trobant parelles freqüents

- El problema més dur és normalment el de trobar les parelles més freqüents.
  - Sovint, les parelles freqüents són bastant nombroses. Les tripletes freqüents són molt més rares. Si tenim $n$ elements hi ha $\binom{n}{r} = \frac{n!}{k!(n-k)!}$ possibles conjunts de $k$ elements. Per $n = 20$ tenim 190 possibles parelles.
  - La probabilitat de que un conjunt d'ítems sigui freqüent baixa de forma exponencial respecte a la mida del conjunt.
- Per tant, ens concentrarem en les parelles i després ho estendrem a conjunts més nombrosos.

# Un algorisme ingenu

- Llegim el fitxer un cop i contem a la memòria principal el nombre d'ocurrències de cada parella.
  - Per cada bossa de $n$ ítems, generem les seves $\frac{n(n-1)}{2}$ parelles amb dos iteracions niuades.
- L'algorisme donarà un error si $(\#items)^2$ excedeix la mida de la memòria.
  - El nombre d'ítems pot ser de l'ordre de 100.000 (en un gran supermercat) o de 10.000.000.000 si parlem de pàgines web.

# Com contar les parelles

- Primera aproximació: Emmagatzemar les tripletes $[i, j, c]$ tals que $count(i, j) = c$.
  - Si els enters i els identificadors dels ítems es poden guardar en 4 bytes, com a mínim necessitem 12 bytes per cada parella.
  - Si ho guardem en un diccionari (**taula hash**) necessitarem una mica més de memòria adicional, però estalviarem l'espai correponent als conjunts amb contador a 0.

# Com contar les parelles

Nota: Podem usar com a clau d'un diccionari la tupla formada per $(i, j)$.

```
i1 = 1
i2 = 2
a = (i1,i2)
a
```

```
(1, 2)
```

```
d = {}
d[a]=23
d[a]
```

```
23
```

# Com contar les parelles

- Què passa si la majoria de parelles surten alguna vegada? Doncs que necessitarem $6n^2$ bytes per guardar-les i això només ens permet tenir un nombre d'ítems de l'ordre de milers en un ordinador convencional.

# Com contar les parelles

- Observació: Podem reduir espai si intentem emmagatzemar $\binom{n}{2}$ contadors de manera que sigui fàcil trobar-los a memòria donada la parella $(i, j)$, sense escriure $(i, j)$ (un ítem pot ser un string).
- Representar els ítems amb enters ho simplificarà.

# Com contar les parelles

- Segona aproximació (**Matriu Triagular**):
  - Numerem els ítems 1,2,3,...
  - Contem $\{i, j\}$ només si $i < j$ i incrementem el valor $A[i, j]$ d'una matriu triangular.

  Amb això estalviem l'espai per guardar els identificadors dels productes i la mitat de la matriu.
- A més a més, si mantenim els contadors de les parelles en ordre lexicogràfic podem optimitzar-ho encara guardant la matriu en una **llista triangular** $A[i]$:
  - $\{1, 2\}, \{1, 3\}, \ldots, \{1, n\}, \{2, 3\}, \{2, 4\}, \ldots \{3, 4\}, \ldots$
  - La parella $\{i, j\}$ està a la posició $A[(i-1)(n - i/2) + j - i]$.
- El nombre total de parelles és $\frac{n(n-1)}{2}$, que ocupen un total de $2n^2$ bytes (4 bytes per parella).

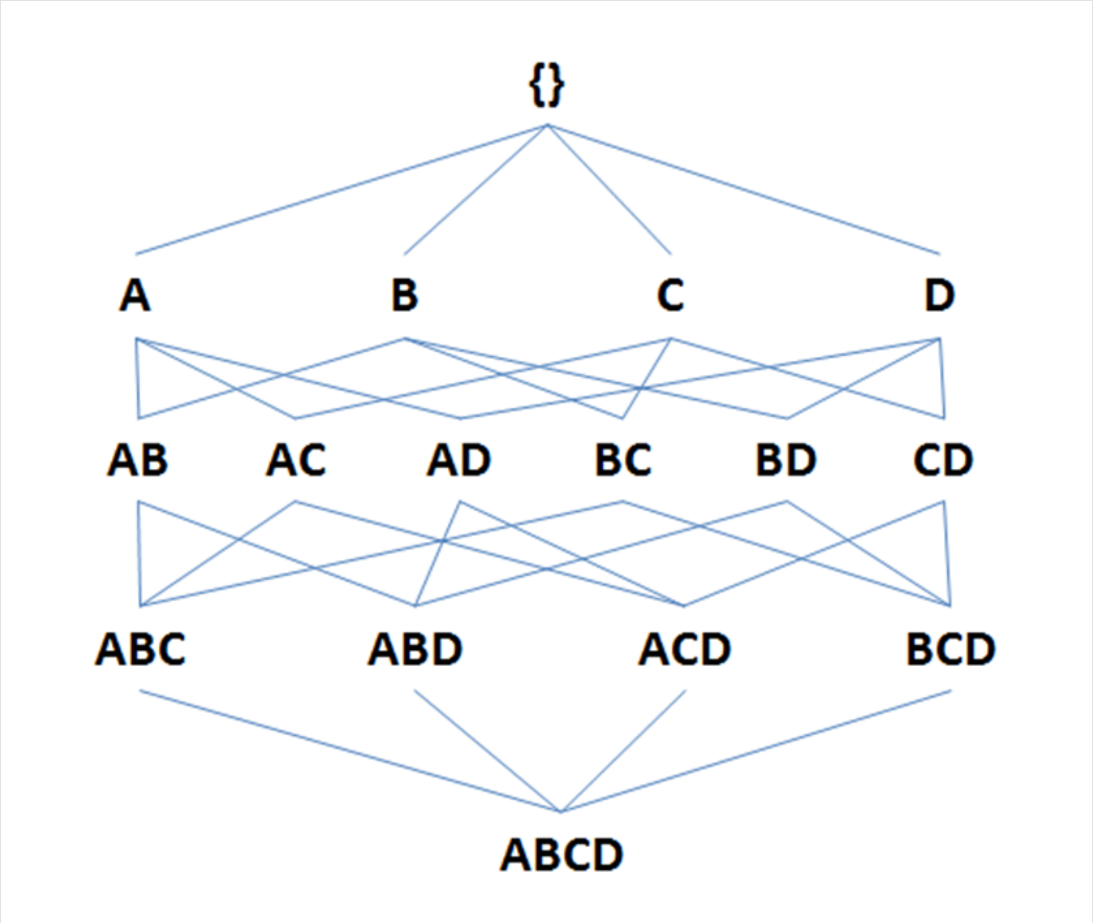
# Com contar les parelles

- El mètode de la matriu triangular (4 bytes per parella) és millor que que una taula hash (12 bytes per parella) si al menys 1/3 de les $\binom{n}{2}$ parelles apareix en alguna bossa. Sinó, són millor les taules hash.

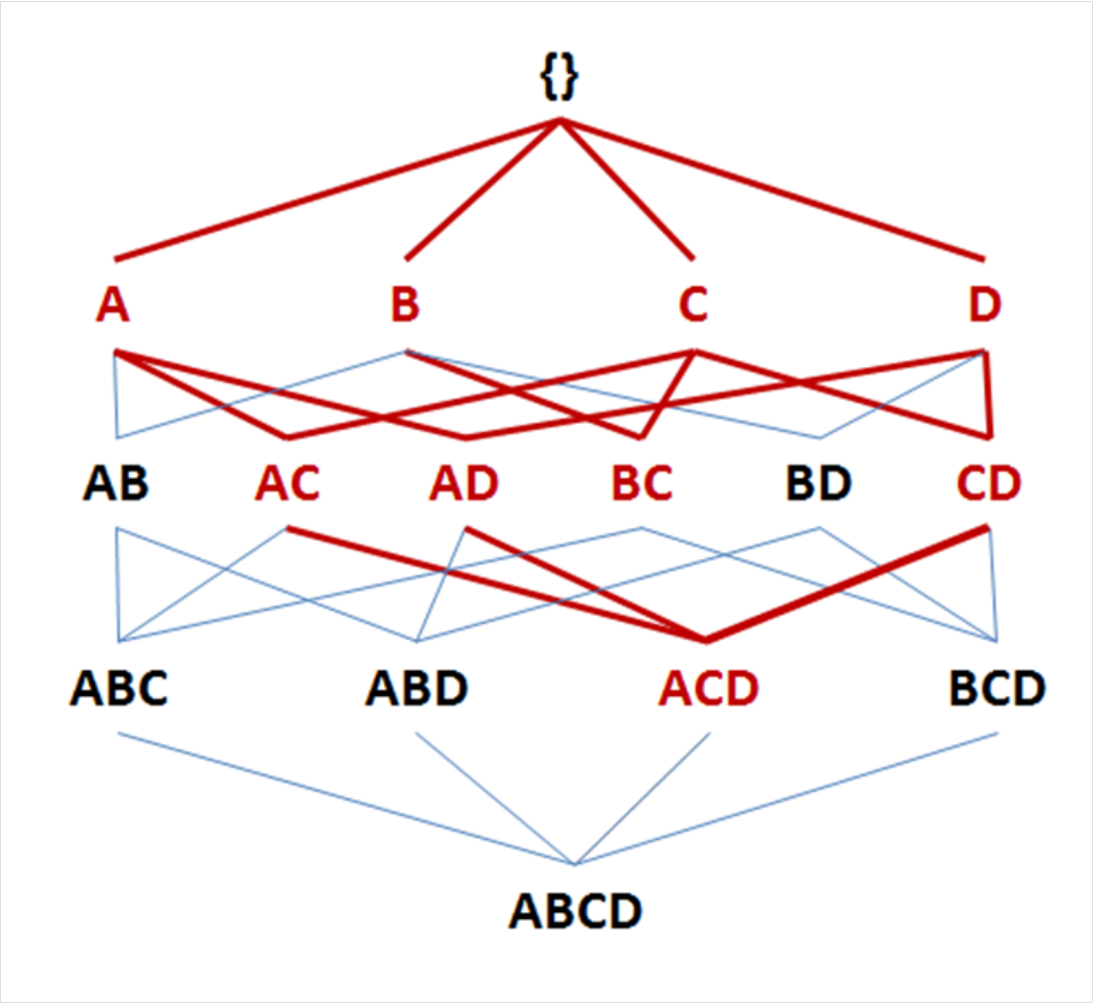
# L'algorisme Apriori

- L'algorisme *Apriori* és un mètode que busca conjunts d'ítems freqüents amb molt poques lectures de les dades i fa un s molt limitat de la memòria.
- La seva idea principal és la *monotonia*: Si un conjunt d'ítems apareix al menys $s$ vegades al conjunt, tots els seus possibles subconjunts també apareixen al menys $s$ vegades!
- Per una altra banda, si un ítem $i$ no apareix a $s$ bosses, llavors cap conjunt que inclogui $i$ pot aparèixer a $s$ bosses.

# L'algorisme Apriori



# L'algorisme Apriori



# L'algorisme Apriori

- A la primera passada llegim les bosses, traduïm els ítems a enters i contem a la memòria principal les ocurrències de cada ítem. Aquest procés requereix una part de la memòria proporcional al nombre d'ítems.
- Després identifiquem els ítems que apareixen al menys $s$ vegades, i per tant són els *ítems freqüents*.

# L'algorisme Apriori

- A la segona passada, llegim les bosses una altra vegada, posem a 0 els contadors dels ítems no freqüents i contem només les parelles per les que els dos elements van ser etiquetats com a freqüents a la passada anterior.
  - Per la propietat de monotonia segur que no perdem cap parella freqüent.
  - Requereix una quantitat de memòria proporcional al quadrat dels ítems freqüents (per guardar els contadors) $O(m^2)$.
  - També necessita una llista dels ítems freqüents (per saber què hem de contar)

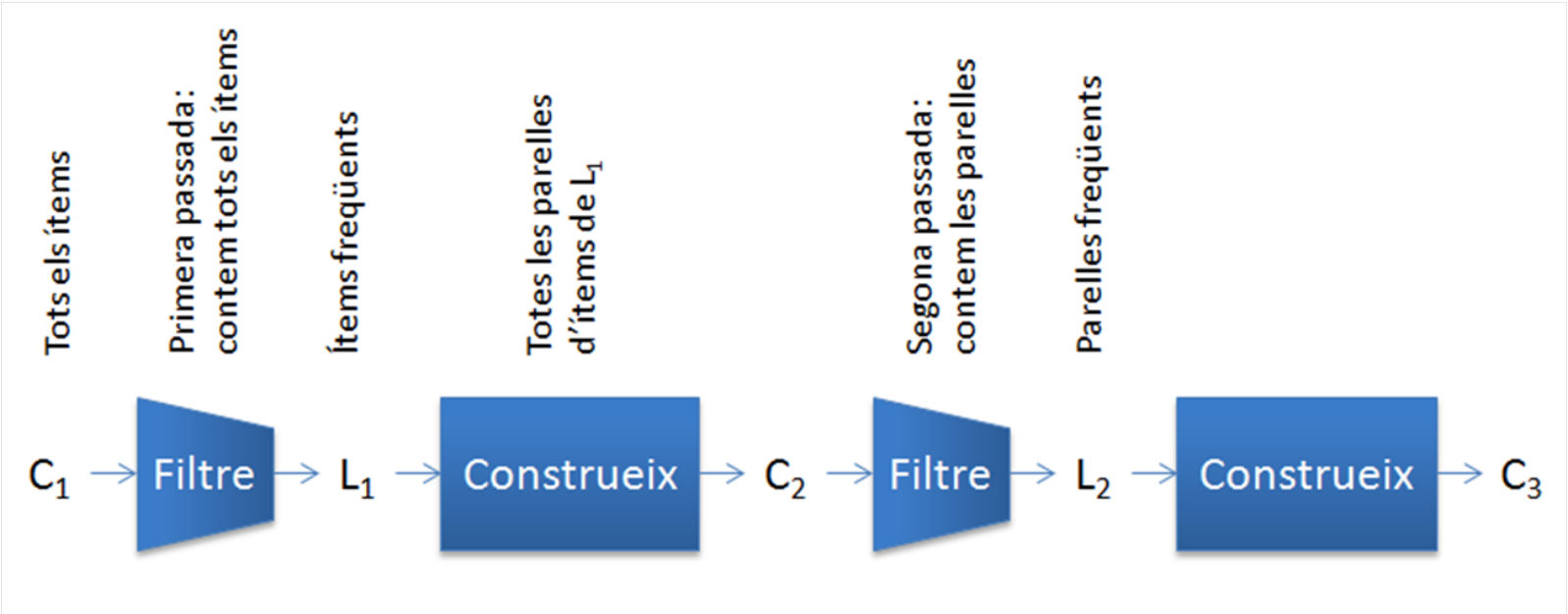
# L'algorisme Apriori

- Podem fer servir el mètode de la matriu triangular amb un $m$ corresponent al nombre d'ítems freqüents.
- Per fer-ho, renumerem els ítems freqüents $1, 2, \ldots$ i mantenim una taula que relaciona els nous nombres amb els identificadors originals dels ítems.

# Tripletes, etc.

- Per construir els conjunts de $k$ ítems a partir dels de $k-1$, construim dos $k$-conjunts (conjunts de mida $k$) a partir dels anteriors:
  - $C_k$ = $k$-conjunts candidats = aquells conjunts que poden ser freqüents, basant-se en la informació de la passada per $k-1$.
  - $L_k$ = el conjunt de $k$-conjunts realment freqüents.

# L'algorisme Apriori



# Apriori per tots els conjunts d'ítems freqüents

- Un pas per cada $k$.
- Necessita espai a la memòria principal per contar cada $k$-conjunt candidat.
- Per un conjunt de dades típic i un suport raonable (p.e., 1%), $k = 2$ és el pas que necessita més memòria.

# Exemple

| | |
|---|---|
| 001 | I1, I2, I5 |
| 002 | I2, I4 |
| 003 | I2,I3 |
| 004 | I1,I2,I4 |
| 005 | I1, I3 |
| 006 | I2, I3 |
| 007 | I1, I3 |
| 008 | I1, I2 ,I3, I5 |
| 009 | I1, I2, I3 |

- Considerem una base de dades, $D$, amb 9 transaccions.
- Suposem que el suport mínim és 2.
- Definim el nivell de confiança mínim com el 70%.
- L'objectiu és generar les regles d'associació.

# Exemple

Pas 1: Crear els 1-conjunts més freqüents.

Llegim $D$ per contar cada un desl candidats i creem $C_1$:

| Conjunts | Contador |
|---|---|
| {I1 } | 6 |
| {I2 } | 7 |
| {I3 } | 6 |
| {I4 } | 2 |
| {I5 } | 2 |

Comparem el suport de cada candidat amb el valor de mínim suport i creem $L_1$:

| Conjunts | Contador |
|---|---|
| {I1 } | 6 |
| {I2 } | 7 |
| {I3 } | 6 |
| {I4 } | 2 |
| {I5 } | 2 |

# Exemple

Pas 2: Generar els 2-conjunts més freqüents.

Generem els candidats $C_2$ des de $L_1$ i contem quantes vegades apareixen:

| Conjunts | Contador |
|---|---|
| {I1, I2 } | 4 |
| {I1, I3 } | 4 |
| {I1, I4 } | 1 |
| {I1, I5 } | 2 |
| {I2, I3 } | 4 |
| {I2, I4 } | 2 |
| {I2, I5 } | 2 |
| {I3, I4 } | 0 |
| {I3, I5 } | 1 |
| {I4, I5 } | 0 |

Comparem el suport de cada candidat amb el valor de mínim suport i creem $L_2$:

| Conjunts | Contador |
|---|---|
| {I1, I2 } | 4 |
| {I1, I3 } | 5 |
| {I1, I5 } | 2 |
| {I2, I3 } | 4 |
| {I2, I4 } | 2 |
| {I2, I5 } | 2 |

# Exemple

Pas 3: Generar els 3-conjunts més freqüents.

Generem els candidats $C_3$ des de $L_2$ i contem quantes vegades apareixen:

| Conjunts | Contador |
|---|---|
| {I1, I2, I3 } | 2 |
| {I1, I2, I5 } | 2 |

Comparem el suport de cada candidat amb el valor de mínim suport i creem $L_3$:

| Conjunts | Contador |
|---|---|
| {I1, I2, I3 } | 2 |
| {I1, I2, I5 } | 2 |

- La generació dels 3-conjunts més freqüents ha fet s de la propietat de l'algorisme Apriori.
- Aquesta propietat també ens diu que cap subconjunt de 4 candidats pot ser freqüent.
- Observació: {I2, I3, I5 } no hi és perquè {I3, I5 } no hi és a $L_2$.

# Exemple: Regles d'associació

- Per cada conjunt d'ítem freqüent $I$ generarem tots els subconjunts no buits de $I$.
- Per cada subconjunt no buit $s$ de $I$ produirem una regla $s \rightarrow (I - s)$ si $\frac{suport(I)}{suport(s)} \geq$ confiança-mínima.
- El resultat final és que hem trobat com a conjunts d'ítems freqüents: {I1}, {I2}, {I3}, {I4}, {I5}, {I1,I2}, {I1,I3}, {I1,I5}, {I2, I3}, {I2, I4}, {I2,I5}, {I1,I2,I3}, {I1,I2,I5} .
- Si considerem {I1,I2,I5}, els seus subconjunt possibles són {I1}, {I2}, {I5}, {I1,I2}, {I1,I5}, {I2,I5} .

# Exemple: Regles d'associació

- Suposem que el nivell mínim de confiança és del 70%.
- Les cerca de regles d'associació és:
  - I1 $\wedge$ I2 $\rightarrow$ I5, Confiança=2/4=50%, regla rebutjada.
  - I1 $\wedge$ I5 $\rightarrow$ I2, Confiança=2/2=100%, **regla acceptada**.
  - I2 $\wedge$ I5 $\rightarrow$ I1, Confiança=2/2=100%, **regla acceptada**.
  - I1 $\rightarrow$ I2 $\wedge$ I5, Confiança=2/6=33%, regla rebutjada.
  - I2 $\rightarrow$ I1 $\wedge$ I5, Confiança=2/7=29%, regla rebutjada.
  - I5 $\rightarrow$ I1 $\wedge$ I2, Confiança=2/2=100%, **regla acceptada**.