# Fouille de Données et Apprentissage

## C3 - Règles d'association

Lina Soualmia

Université de Rouen
LITIS - Équipe TIBS-CISMeF
lina.soualmia@chu-rouen.fr

1 février 2016

# Crédits

- P.Poncelet (Montpellier)
- S.Ullman (Stanford)
- R.Rakotomalala (Lyon)
- N.Pasquier (Nice)
- parmi d'autres ...

# Plan

- Extraction de motifs fréquents
- Extraction de règles d'association
- Autres algorithmes

# Extraction de motifs fréquents

# Motivation

- Son objectif est de trouver des *motifs* qui apparaissent *fréquemment* dans une base de données
- La version de base de l'extraction de motifs fréquents permet de faire la fouille dans une relation (table) d'une base de données relationnelle dont les valeurs sont des booléens qui indiquent la présence ou l'absence d'une propriété.
- Définition : une base de données formelle est la donnée d'un triplet $(O, P, R)$ où :
  - $O$ est un ensemble fini d'objets
  - $P$ est un ensemble de propriétés
  - $R$ est une relation sur $O \times P$ qui permet d'indiquer si un objet $x$ a une propriété $p$ (noté $x\ R\ p$).

# Exemple

| R | a | b | c | d | e |
|---|---|---|---|---|---|
| $x_1$ | 1 | 0 | 1 | 1 | 0 |
| $x_2$ | 0 | 1 | 1 | 0 | 1 |
| $x_3$ | 1 | 1 | 1 | 0 | 1 |
| $x_4$ | 0 | 1 | 0 | 0 | 1 |
| $x_5$ | 1 | 1 | 1 | 0 | 1 |
| $x_6$ | 0 | 1 | 1 | 0 | 1 |

- $O = \{x_1, x_2, x_3, ..., x_6\}$
- $P = \{a, b, c, d, e\}$
- $x\ R\ p$ si et seulement si la ligne de $x$ et de $p$ se croisent sur un 1 : $x_1\ R\ a$

# Autre format possible

| R | *items* |
|---|---|
| $x_1$ | acd |
| $x_2$ | bce |
| $x_3$ | abce |
| $x_4$ | be |
| $x_5$ | abce |
| $x_6$ | bce |

Table avec attribut multi-valué

## Définition d'un motif

- Un motif d'une base de donnée formelle $(O, P, R)$ est un sous-ensemble de $P$.
- L'ensemble de tous les motifs d'une base est donc l'ensemble des parties de $P$ noté $2^P$
- On dira qu'un objet $x \in O$ possède un motif $m$ si $\forall\, p \in m,\ x\, R\, p$

## Exemples de motifs

Pour la base on a $2^P = 2^5 = 32$ motifs

# Motifs fréquents

- Parmi cet ensemble de motifs, on va chercher ceux qui apparaissent fréquemment.
- Pour chercher les motifs fréquents on a besoin d'introduire les notions de *connexion de Galois* et de *support* d'un motif.

# Connexion de Galois

- La connexion de Galois associée à une base de données formelle $(O, P, R)$ est le couple de fonctions $(f, g)$ définies par :
- $f : 2^P \longrightarrow 2^O$
  $m \longmapsto f(m) = \{x \in O \mid x \text{ possède } m\}$
- $g : 2^O \longrightarrow 2^P$
  $X \longmapsto g(X) = \{p \in P \mid \forall x \in X, x R p\}$
- $g$ est dit dual de $f$ et $f$ dual de $g$.
- $f(m)$ est l'image du motif $m$.

# Support d'un motif

- Soit $m \in 2^P$ un motif. Le support de $m$ est la proportion d'objets dans $O$ qui possèdent le motif :
  - $support : 2^P \longrightarrow [0;1]$
    $m \longmapsto support(m) = \frac{|f(m)|}{|O|}$
  - exemple $support(a) = 3/6$
- Le support est décroissant
- Si $m$ est un sous-motif de $m'$ $(m \subseteq m')$ alors $support(m) \geq support(m')$

# Motif fréquent

- Le support mesure la fréquence d'un motif :
  - plus il est élevé, plus le motif est fréquent.
- On distinguera les motifs *fréquents* des motifs non fréquents à l'aide d'un seuil $\sigma_s$
- Soit $\sigma_s \in [0; 1]$. Un motif $m$ est fréquent si $support(m) \geq \sigma_s$ sinon il est dit non fréquent.

# Extraction des motifs fréquents

- Une approche naïve pour l'extraction des motifs fréquents consiste à
  - parcourir l'ensemble $2^P$ de tous les motifs,
  - à calculer leurs supports
  - et à ne garder que les plus fréquents.
- Malheureusement cette approche est trop consommatrice en temps : le nombre de motifs est $2^{|P|}$ et en pratique on veut manipuler des bases ayant un grand nombre de propriétés
- L'approche que nous allons décrire permet d'extraire des motifs fréquents dans une base ayant plusieurs milliers de propriétés et plusieurs millions d'objets

# Extraction par niveaux

- L'extraction par niveaux s'appuie sur la décroissance du support et se fait selon le principe suivant :

1. On commence par chercher les motifs fréquents de longueur 1 ;
2. On combine ces motifs pour obtenir des motifs de longueur 2 et on ne garde que les fréquents parmi eux ;
3. On combine ces motifs pour obtenir des motifs de longueur 3 et on ne garde que les fréquents parmi eux ;
4. ...etc.

# Extraction par niveaux

- Tout sous-motif d'un motif fréquent est fréquent.
  - si $m' \subseteq m$ et $support(m) \geq \sigma_S$ alors $support(m') \geq \sigma_S$
- Tout super-motif d'un motif non fréquent est non fréquent
  - si $m' \supseteq m$ et $support(m) < \sigma_S$ alors $support(m') < \sigma_S$

`Apriori`

**entrées :** $(\mathcal{O}, \mathcal{P}, \mathcal{R})$ : une base de données formelle
$\sigma_s \in [0; 1]$

**sortie :** l'ensemble des motifs fréquents de la base, relativement au seuil $\sigma_s$.

**Début**

$i \leftarrow 1$

$\mathtt{C}_1 \leftarrow$ ensemble des motifs de taille 1

**tant que** $\mathtt{C}_i \neq \emptyset$ **faire**

Calculer le support de chaque motif $m \in \mathtt{C}_i$

$\mathtt{F}_i \leftarrow \{m \in \mathtt{C}_i \mid \mathtt{support}(m) \geqslant \sigma_s\}$

$\mathtt{C}_{i+1} \leftarrow \mathtt{générer\text{-}candidats}(\mathtt{F}_i)$

$i \leftarrow i + 1$

**fin-tant que**

**retourner** $\bigcup_{i \geqslant 1} \mathtt{F}_i$

**Fin**

Déroulement de l'algorithme Apriori sur l'exemple avec $\sigma_S = 2/6$

| R | a | b | c | d | e |
|---|---|---|---|---|---|
| $x_1$ | 1 | 0 | 1 | 1 | 0 |
| $x_2$ | 0 | 1 | 1 | 0 | 1 |
| $x_3$ | 1 | 1 | 1 | 0 | 1 |
| $x_4$ | 0 | 1 | 0 | 0 | 1 |
| $x_5$ | 1 | 1 | 1 | 0 | 1 |
| $x_6$ | 0 | 1 | 1 | 0 | 1 |

# Génération des candidats

Exemple :

- i = 4
- 3-itemsets fréquents : {abc, abd, acd, ace, bcd}
- Jointure des 3-itemsets de même préfixe
  - abc et abd : abcd
  - acd et ace : acde
- Élagage : *acde* est supprimé car *ade* n'est pas dans l'ensemble des 3-itemsets fréquents
- $C_4$= {abcd}

## Optimisation d'Apriori

- Inconvénient : N passes sur la base :
  - une pour 1, 2, ... N-ensembles, N étant la taille du plus grand ensemble fréquent
  - comptage des ensembles fréquents par transactions en parcourant la table
  - Trouver les produits d'une transaction peut nécessiter de la mémoire si la table est normalisée

## Partition

- But : Réduire le nombre de passes
- Principe :
  - partitionner la base de manière à ce que chaque partition tienne en mémoire centrale (utilisation d 'Apriori pour chaque partition)
  - 2 passes sur la base

## Partition

- Phase 1 : Division de la base
  - Traiter les partitions une par une : les itemsets fréquents sont fusionnés pour générer l'ensemble de tous les itemsets fréquents potentiels
- Phase 2 : le support de ces itemsets est calculé

# Partition [95]

- La base est divisée en N partitions
  - chaque partition tient en mémoire
- Chaque partition est traitée indépendamment
  - découverte des ensembles fréquents pour chaque partition
- Remarque : un ensemble fréquent doit l'être dans au moins une partition
- Exploration de l'union des ensembles fréquents sur la base
  - comptage en une passe
  - élimination des ensembles non fréquents
- Avantage :
  - deux passes au plus
  - parallélisable facilement

## Dynamic Itemset Counting

- But : réduction du nombre de balayages de la base
- Lecture par blocs de M transactions
- Essayer de générer le plus vite possible, i.e. à la fin de M, des (i+1)-itemsets pour les rechercher dans les prochaines M transactions

## MaxMiner : Mining Max-patterns [97]

- But : rechercher les plus longs itemsets fréquents
- Max-patterns : bordures de motifs fréquents
  - Un sous-ensemble d'un max-pattern est fréquent
  - Un sur-ensemble d'un max-pattern est non fréquent
- Parcours en largeur et en profondeur (voir exple)

# Algorithme FP-Growth [00]

- Représentation des items fréquents par un index spécial FP-tree (Frequent Pattern Tree)
- Construction du FP-tree
  - Déterminer les items fréquents (1-ens.)
  - Trier par fréquence décroissante (table)
  - Créer une racine vide
  - Pour chaque transaction :
    - ajout des chemins de produits ordonnés par fréquence
    - fusion des sous-chemins communs
    - mise à jour des compteurs de fréquence des produits
  - Générer les ensembles fréquents par combinaison des nœuds des chemins fréquents

(voir exple)

# Extraction des règles d'association

# Définition

- Les règles d'association sont de la forme : $r = p_1 \rightarrow p_2$ où $p_1$ et $p_2$ sont deux motifs
- Une telle règle peut se lire intuitivement :
  - Si un objet $x$ possède $p_1$ alors il est plausible que $x$ possède $p_2$.
- On s'appuie également sur la notion de base de données formelle (O, P, R)
- Une règle d'association $r$ est la donnée de deux motifs A et B. on la note $r = A \rightarrow B$
- A est appelé antécédent de la règle et B son conséquent

# Support et confiance

- Le support d'une règle d'association $r = A \rightarrow B$ est :
  - $support(A \rightarrow B) = support(A \cup B)$
- La confiance d'association $r = A \rightarrow B$ est :
  - $confiance(A \rightarrow B) = \frac{support(A \cup B)}{support(A)}$
  - $confiance(r) \in [0; 1]$
  - Elle indique la proportion des objets qui possèdent à la fois A et B parmi les objets qui possèdent A
  - $confiance(A \rightarrow B) = 1$ : tous les objets qui possèdent le motif A possèdent le motif B
  - $confiance(A \rightarrow B) = 0$ : signifie que les objets possédant le motif A ne possèdent pas le motif B.

## Remarque

- On peut rapprocher le support d'une règle d'une probabilité :
  - $support(A \rightarrow B)$ est la probabilité que x possède le motif A **et** que x possède le motif B .
  - Propriété : $support(A \rightarrow B) = support(B \rightarrow A)$
- On peut rapprocher la confiance des probabilités conditionnelles :
  - $confiance(A \rightarrow B)$ est la probabilité que x possède le motif B sachant que x possède le motif A

# Règle valide

- Parmi les règles on s'intéresse aux règles dites valides :
- Une règle d'association $r$ est valide si elle vérifie :
  - $support(r) \geq \sigma_S$
  - $confiance(r) \geq \sigma_C$
  - $\sigma_S$, $\sigma_C \in [0; 1]$ sont les valeurs de seuil prédéfinies

# Règle exacte ; Règle approximative

- Une règle d'association est exacte si $confiance(r) = 1$
- Dans le cas contraire elle est dite approximative
- Si $r = A \rightarrow B$ est exacte alors $support(A \cup B) = support(A)$
- Ex : $support(a) = support(ac) = \frac{3}{6}$; $a \rightarrow c$ est une règle exacte
- Mais : ce n'est pas parce qu'une règle est exacte qu'elle est certaine :
  - cela signifie seulement qu'il n'existe aucun contre-exemple dans la base de données fouillée à cette règle

# Remarque

- Si on a par exemple $ab \rightarrow ac$ on peut noter qu'elle a même support et même confiance que la règle $ab \rightarrow c$
- Répéter en partie droite d'une règle une propriété apparaissant en partie gauche est inutile.
- De façon générale quand on a une règle $A \rightarrow B$ alors $A \cap B = \phi$
- On pe=ut noter les règles sous la forme $r = p_1 \rightarrow p_2 \backslash p_1 \text{avec } p_1 \subseteq p_2$
- $E \backslash F = \{x \mid x \in E \text{ et } x \notin F\}$
- Par exemple pour $ab \rightarrow c$ on aura $p_1 = ab$ et $p_2 = abc$

# Support et confiance de r

- $r = p_1 \rightarrow p_2 \backslash p_1$
- $support(r) = support(p_2)$
  - $ab \rightarrow c$ ; $support(r) = support(abc)$
- $confiance(r) = support(p_2)/support(p_1)$
  - $ab \rightarrow c$ ; $confiance(r) = support(abc)/support(ab)$

# Propriétés des règles d'association

- Pas de composition des règles :
  - Si X→Z et Y→Z sont vrais dans la base, XY →Z n'est pas nécessairement vrai
- Pas de décomposition des règles :
  - Si XY →Z convient, X→Y et Y→Z peut ne pas être vrai
- Pas de transitivité
  - Si X→Y et Y→Z sont vrais dans la base, nous ne pouvons pas en déduire que X →Z

# Schéma algorithmique de base

- La plupart des approches utilisent le même schéma algorithmique
- Pour construire les règles d'association, le support de tous les motifs fréquents dans la base doit être calculé
- L'algorithme procède en 2 phases :

1. Génération de tous les ensembles fréquents
2. Génération des règles d'association

# Algorithme de génération

Le principe est le suivant :

- On s'intéresse aux règles valides r telles que $support(r) = support(p_2) \geq \sigma_S$

1. On considère les règles de la forme $p_1 \rightarrow p_2 \backslash p_1$ où la conclusion est de longueur 1
2. On élague les règles non valides
3. On combine les conclusions des règles valides
4. Puis on passe à des conclusions de longueur 2 et on itère (longueurs 3, 4, etc. . . )

- L'exemple montre que cela fait beaucoup de règles : problème d'intelligibilité pour l'analyste

1. Une stratégie d'élagage consiste a définir de nouvelles mesures de plausibilité (autres que le support et la confiance) pour limiter cet ensemble de règles.
2. Une deuxième stratégie consiste à élaguer les règles déjà contenues dans la base de connaissances.
   - Si, par exemple, on dispose de l'ontologie du cours 1, et qu'on a extrait la règle `tortue`→`quadrupède`, `reptile`, cette règle peut être supprimée de l'ensemble des règles valides puisqu'elle n'apporte pas de nouvelles connaissances.
3. Une troisième stratégie consiste à constater que certaines règles sont moins informatives que d'autres.
   - Par exemple, si on a les règles $R_1$= ab →c et $R_2$= a →cd avec le même support et la même confiance, alors, on pourra élaguer $R_1$, qui est une conséquence de $R_2$.

# Autres mesures

**Limites de la confiance :**

- Cas particulier : si la conséquence de la règle très fréquente, cela implique une confiance élevée
- Exemple : jeu de 10,000 transactions de vente
  - support(livre) = 75% ; P(livre) = 0,75
  - support(DVD) = 60% ; P(DVD) = 0,6
  - support(livre, DVD) = 40% , P(livre, DVD) = 0,4
  - DVD $\rightarrow$ livre :
    - support = 40%, confiance= 66% ; P(livre | DVD) = 0,66
  - Personnes achetant un DVD : 66% achètent un livre
  - Toutes les personnes : 75% achètent du livre

## Table de contingence

- Comptage des effectifs

| | Livre | ¬Livre | $\sum$ |
|---|---|---|---|
| DVD | 4000 | 2000 | 6000 |
| ¬DVD | 3500 | 500 | 4000 |
| $\sum$ | 7500 | 2500 | 10000 |

- Utilité :
  - calcul des mesures
  - poids des itemsets
  - répartition des valeurs

## Mesure du Lift (intérêt)

- Lift : permet de tenir compte de la fréquence de la conséquence
- $lift = \frac{P(antécédent+conséquence)}{P(antécédent)\times P(conséquence)} \in [0, +\infty[$
- Mesure la corrélation statistique entre antécédent et conséquence
  - $lift = 1 : P(AC) = P(A) \times P(C) \equiv indépendance$
  - $lift < 1 : P(AC) < P(A) \times P(C) \equiv corrélation\ négative$
  - $lift > 1 : P(AC) > P(A) \times P(C) \equiv corrélation\ positive$
- les règles de $lift \leq 1$ sont inintéressantes

**Sur l'exemple :**

- DVD $\rightarrow$ livre : support $= 40\%$, confiance$= 66\%$
- $lift = \frac{P(DVD,livre)}{P(DVD)\times P(livre)} = \frac{0.4}{0.6\times 0.75} = 0.89 < 1$
- la règle est inintéressante
- Il est nécessaire
  - soit d'observer la répartition des valeurs (support) de la conséquence de la règle
  - soit d'utiliser une mesure de corrélation (comme le lift) pour supprimer les règles inintéressants (statistiquement)
- La confiance reste une mesure utile associée au lift

## Mesure de la conviction

- La conviction tient compte de l'absence de la conséquence
- $conviction = \frac{P(antécédent) \times P(\neg conséquence)}{P(antcédent + \neg conséquence)} \in [0, +\infty[$
- Mesure la corrélation statistique entre la présence de l'antécédent et l'absence de la conséquence :
  - $conviction = 1 : P(A) \times P(\neg C) = P(A\neg C) \equiv indépendance$
  - $conviction > 1 : P(A) \times P(\neg C) > P(A\neg C) \equiv$ *corrélation positive*
  - $conviction < 1 : P(A) \times P(\neg C) < P(A\neg C) \equiv$ *corrélation négative*
- Les règles de $conviction \leq 1$ sont inintéressantes

Exemple (suite)

- DVD $\rightarrow$livre : support = 40%, confiance= 66%
- support(DVD,$\neg$livre) = support(DVD) – support(DVD, livre) = 60% - 40% = 20%
- P(DVD, $\neg$livre) = 0,20
- support($\neg$livre) = 100% - support(livre) = 100% - 75% = 25% ; P($\neg$livre) = 0,25
- conviction = P(DVD) x P($\neg$livre) / P(DVD, $\neg$livre) = (0,6 x 0,25) / 0,2 = 0,75
- Conviction inférieure à 1 : règle inintéressante

## Limites du Lift et de la Conviction

- Mesures non directionnelles :
  - lift(A→C) = lift(C→A)
  - conviction(A →C) = conviction(C→A)
- Sensibles au nombre de lignes du jeu de données

| Jeu de données | AC | ¬AC | A¬C | ¬A¬C | Conf | lift | conv |
|---|---|---|---|---|---|---|---|
| A1 | 1000 | 100 | 100 | 100000 | 91% | 83,64 | 10,88 |
| A2 | 1000 | 100 | 100 | 10000 | 91% | 9,26 | 9,92 |
| A3 | 1000 | 100 | 100 | 1000 | 91% | 1,82 | 5,50 |
| A4 | 1000 | 100 | 100 | 0 | 91% | 0,99 | 0,92 |

Mesures non null-invariant : nombre de lignes ¬A¬C influe sur la mesure

## Mesures complémentaires

- Cosine : permet de supprimer l'influence du nombre total de lignes sur le résultat
  - $cosine = \frac{P(antécédent+conséquence)}{\sqrt{P(antécédent)\times P(conséquence)}} \in [0,1[$
  - $cosine = \frac{count(antécédent+conséquence)}{\sqrt{count(antécédent)\times count(conséquence)}}$
  - cosine=0,5 : indépendance
  - cosine>0,5 : corrélation positive
  - cosine<0,5 : corrélation négative
- Les règles de cosine $\leq$0,5 sont inintéressantes

## Null-invariant Cosine

- Lignes $\neg A \neg C$ n'influent pas sur la mesure

| | AC | $\neg$AC | A$\neg$C | $\neg$A$\neg$C | Conf | lift | conv | cosine |
|---|---|---|---|---|---|---|---|---|
| A1 | 1000 | 100 | 100 | 100000 | 91% | 83,64 | 10,88 | 0,91 |
| A2 | 1000 | 100 | 100 | 10000 | 91% | 9,26 | 9,92 | 0,91 |
| A3 | 1000 | 100 | 100 | 1000 | 91% | 1,82 | 5,50 | 0,91 |
| A4 | 1000 | 100 | 100 | 0 | 91% | 0,99 | 0,92 | 0,91 |

- Rmq : sur cet exemple, la confiance et cosine donnent la même valeur car count(A)=count(C)=1100
- Inconvénient : les valeurs proches de 0,5 ne permettent pas de déterminer une corrélation positive entre A et C

## Limites du Cosine

Exemple (suite)

- DVD $\rightarrow$livre : support $= 40\%$, confiance$= 66\%$
- count(livre, DVD) $= 4000$ , count(livre) $= 7500$, count(DVD) $= 6000$
- cosine $= 0{,}60 > 0{,}5$ : règle supposée intéressante
- Mais cette règle ne l'est pas (cf. Lift et Conviction)

# Quelles mesures utiliser ?

- Propriété des mesures

| Mesure | corrélation | null-invariant | fréq.Consequence | directionnelle |
|---|---|---|---|---|
| confiance | N | O | N | O |
| lift | O | N | O | N |
| conviction | O | N | O | N |
| cosine | O | O | N | N |

# Quelles mesures utiliser ?

- Chaque propriété précise l'interprétation
  - Directionnelle : distingue les liens A $\rightarrow$C et C $\rightarrow$A
  - Corrélation : validité statistique
  - Null-invariant : indépendamment des autres lignes
- Support nécessaire : taille de la population concernée
- Quelles mesures ?
  - Optimal : une mesure pour chaque propriété
  - Minimum : support, confiance, lift (ou conviction)

- Ne garder que les règles les plus *informatives*
- Pour un motif donné, les règles valides les plus intéressantes sont à conclusions maximales (autrement dit : à prémisses minimales)
- Si on a une règle R = $p_1 \rightarrow p_2 \backslash p_1$ valide et $p'_1$ tel que $p_1 \subseteq p'_1 \subseteq p_2$, alors, avec R' = $p'_1 \rightarrow p_2 \backslash p'_1$ , on a :
  - $support(R') = support(R) = support(p_2)$
  - $confiance(R') = \frac{support(p_2)}{support(p'_1)} \geq \frac{support(p_2)}{support(p_1)} = support(R)$
  - donc si R est valide, R' l'est aussi.

Exemple :

- la règle $e \rightarrow abc$ est valide.
- On peut en déduire que les règles suivantes sont également valides :
  - $ea \rightarrow bc$ , $eb \rightarrow ac$ , $ec \rightarrow ab$ , $abe \rightarrow c$ , $ebc \rightarrow a$ ...etc.

Cette méthode permet d'éliminer la redondance dans les règles

- $R1 : a \rightarrow bc$ et $R2 : ab \rightarrow c$
- $R2$ est redondante par rapport à $R1$ :
  - même support :(abc)
  - conf($R2$)=support(abc)/support(ab)
  - conf($R1$)=support(abc)/support(a)
  - donc conf($R2$)>conf($R1$)
- Plus généralement, pour un k-ensemble fréquent, il suffit d'extérioriser la règle valide de condition minimale (ici $R1$)

## La redondance stricte :

- $R1 : a \rightarrow bcd$ et $R2 : ab \rightarrow c$
- $R2$ est redondante par rapport à $R1$ :
  - conf($R1$)=support(abcd)/support(a)
  - conf($R2$)=support(abc)/support(ab)
  - support(a) $>$ sup(ab) et sup(abcd) $<$ sup(abc)
  - donc conf($R2$)$>$conf($R1$)
- Plus généralement, il suffit de considérer le plus grand k-ensemble fréquent et d'extérioriser la règle valide de condition maximale (ici $R2$)

**Génération des règles revue :**

- Il suffit de retrouver les plus grands ensembles de support $>$ MinSup
- puis d'en extraire les règles de confiance $>$ MinConf ayant une condition maximale
- S'il n'y en a pas on descend le semi-treillis des ensembles fréquents et on itère.

## Algorithme Close

- repose sur l'extraction de générateurs d'itemsets fermés fréquents
- le nombre d'itemsets fermés fréquents est généralement bien inférieur au nombre d'itemsets fréquents

# Rappel : connexion de Galois

- La connexion de Galois associée à une base de données formelle $(O, P, R)$ est le couple de fonctions $(f, g)$ définies par :
- $f : 2^P \longrightarrow 2^O$
  $m \longmapsto f(m) = \{x \in O \mid x \text{ possède } m\}$
- $g : 2^O \longrightarrow 2^P$
  $X \longmapsto g(X) = \{p \in P \mid \forall x \in X, x R p\}$
- $g$ est dit dual de $f$ et $f$ dual de $g$.
- $f(m)$ est l'image du motif $m$.

## Opérateur de fermeture

- La fermeture d'un itemset A est un itemset B tel que B apparaît dans les mêmes objets que A.
- Pour la calculer on utilise les deux fonctions :
  - $f$ : qui associe à un itemset les objets qui le contiennent
  - $g$ : qui associe à un ensemble d'objets les itemsets qu'ils ont en commun
- soit A un itemset : $fermeture(A) = g \circ f(A)$ (voir exemple)

## Algorithme Close

1. Initialisation de l'ensemble des générateurs avec l'ensemble des singletons formés par les items
2. Calcul de la fermeture des générateurs de niveau k et de leur support
3. Ajout des fermetures des générateurs à l'ensemble itemsets fermés fréquents
4. Génération des générateurs de niveau k + 1

## Algorithme Close

- Les générateurs de niveau k + 1 sont obtenus de la même manière que dans l'algorithme Apriori, mais ceux appartenant à la fermeture d'un générateur de niveau k sont supprimés. (voir exple)

## Génération des règles d'association par Close [02]

- L'ensemble des générateurs et de leurs fermés permettent de déduire une base générique de règles exactes (conf=1) :
  - par exemple : si le générateur est {a} et que son fermé est {abc} la règle exacte extraite est $a \rightarrow bc$
- L'ensemble des générateurs, des fermés et de leurs sur-ensembles fermés permettent de déduire une base de règles approximatives (conf<1) :
  - si le générateur est {a}, son fermé {abc} et le sur-ensemble fermé {abcd} la règle approximative extraite est $a \rightarrow bcd$