Fouille de données : notes de cours

Jean Lieber (fortement mais librement inspiré du cours d'Amedeo Napoli)

dernière version : 05/10/07
(version préliminaire modifiée par Adrien Coulet)

Contents

1 Introduction

La fouille de données (*data-mining*) est l'étape centrale du processus d'extraction de connaissances des bases de données (ECBD ou *KDD* pour *Knowledge Discovery in Databases* en anglais). Dans cette introduction, nous allons

rappeler la différence entre la notion de donnée et la notion de connaissances, puis nous définirons les principes de l'ECBD et la position de la fouille de données dans l'ECBD.

1.1 Rappel : bases de données et bases de connaissances

1.1.1 Différence entre données et connaissances

- Métaphore du manuel de biologie écrit en hongrois.
- Données : interprétées pour ce qu'elle sont ($\simeq$ faits),
  connaissances : interprétées par des *raisonnements*.
- Deux mots sur les raisonnements :
  - Déductifs (ex. : Socrate est un homme, tout homme est mortel, *donc* Socrate est mortel),
    s'appuie sur des connaissances sous forme de règles (ex. : homme $\Rightarrow$ mortel) ;
  - Non déductif.

  On se contentera dans ce cours de parler des raisonnements déductifs.
- Les connaissances sont décrites dans une *syntaxe* particulière (un langage)
  et ont une *sémantique* (qui leur donne un sens).
- Exemples :
  - $[0; 3[$:
    syntaxe = chaîne de caractères `"[0 ; 3["`,
    sémantique = ensemble des réels x tels que $0 \leqslant x < 3$.
  - $\texttt{chat}(x)$ **et** $\texttt{dort}(x) \Rightarrow \texttt{ronronne}(x)$:
    syntaxe = la formule elle-même (le texte de la formule),
    sémantique = si x est un chat qui dort, alors x ronronne
    (ou $\mathcal{C} \cap \mathcal{D} \subseteq \mathcal{R}$, $\mathcal{C}$ étant l'ensemble des chats, $\mathcal{D}$ étant l'ensemble des dormeurs et $\mathcal{R}$ étant l'ensemble des ronronneurs).

1.1.2 Différences entre bases de données et bases de connaissances

- Différence : quand on interroge une base de données, on n'obtient que des informations qui y sont déjà de façon explicite, quand on interroge une base de connaissances, on peut obtenir des informations qui n'y sont pas explicitement.
- Exemple de base de connaissances :

Règles : $\texttt{père}(x, y) \Rightarrow \texttt{parent}(x, y)$ $\quad$ $\texttt{mère}(x, y) \Rightarrow \texttt{parent}(x, y)$

$\texttt{père}(x, y)$ **et** $\texttt{parent}(y, z) \Rightarrow \texttt{grand-père}(x, z)$

$\texttt{mère}(x, y)$ **et** $\texttt{parent}(y, z) \Rightarrow \texttt{grand-mère}(x, z)$

$\texttt{parent}(x, y) \Rightarrow \texttt{ascendant}(x, y)$ $\quad$ $\texttt{parent}(x, y)$ **et** $\texttt{ascendant}(y, z) \Rightarrow \texttt{ascendant}(x, z)$

Faits : $\texttt{père}(\texttt{Marcel}, \texttt{Maurice})$ $\quad$ $\texttt{mère}(\texttt{Henriette}, \texttt{Maurice})$ $\quad$ $\texttt{père}(\texttt{Maurice}, \texttt{Juliette})$

$\texttt{père}(\texttt{Maurice}, \texttt{Léon})$ $\quad$ $\texttt{mère}(\texttt{Cunégonde}, \texttt{Léon})$

- Exemple de requêtes sur cette base de connaissances :

$\texttt{grand-père}(\texttt{Marcel}, \texttt{Léon})$ $\quad$ $\texttt{grand-père}(?, \texttt{Juliette})$ $\quad$ $\texttt{ascendant}(?, \texttt{Léon})$

$\texttt{ascendant}(\texttt{Henriette}, ?)$ $\quad$ $\texttt{ascendant}(?, ?)$

- La limite est parfois un peu imprécise entre bases de données et bases de connaissances (p. ex., “bases de données déductives”).

Exemple de bases de connaissances : les *ontologies*. On peut les définir informellement de la façon suivante (encore qu'il y ait débat sur cette définition). Une ontologie décrit un domaine par un ensemble de concepts de ce domaine et par les liens entre ces concepts. Parmi ces liens, le plus fréquemment utilisé est le lien de généralité entre concepts, qui indique qu'un concept A est plus général qu'un concept B, ce qui signifie que tous les individus dénotés par A le sont par B. Les concepts sont alors organisés en hiérarchie. La figure 1 donne un exemple simple d'ontologie.

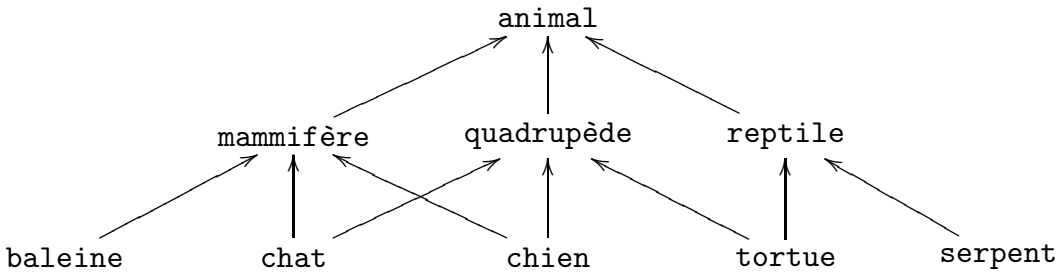


Figure 1: Une ontologie de la zoologie.

1.2 L'extraction de connaissances des bases de données (ECBD)

1.2.1 Origine

- Des années 60 à la fin des années 80 : Développement de l'informatique et des bases de données : système de gestion de bases de données relationelles, méthodes de modélisation , optimisation des requètes
- Fin des années 80 : situation *data rich but information poor* : Disposition de grandes sources de données inexploitées (les “Data Tombs”), besoin d'outils d'analyse puissants pour aider les décideurs $\mapsto$ naissance de l'informatique décisionelle.
  Les premiers travaux en informatique décisionnelle mènent à l'apparition des entrepôts de données (associés à la technologie OLAP), et la notion d'ECBD.
- Depuis : *from data tombs to golden nuggets* : Les besoins en extraction de connaissances continuent d'aumenter avec, entre autre, le développement du Web, le séquençage du génome humain.

1.2.2 Définition

L'ECBD peut être considérée comme un processus d'extraction de connaissances *nouvelles*, potentiellement *utiles* et ayant un degré de *plausibilité*, dans de *grands volumes* de données.

- nouvelles : c'est-à-dire pas déjà connues (on extraira parfois des connaissances qui ne sont pas nouvelles, mais ce n'est pas l'objectif de l'ECBD) ;
- utiles : réutilisable dans un processus de raisonnement ;
- plausibles : on cherche à controller la plausibilité des connaissances extraites ;
- grands volumes de données :
  - nécessitent des processus automatiques ;
  - permet une certaine “validité statistique” des connaissances extraites.

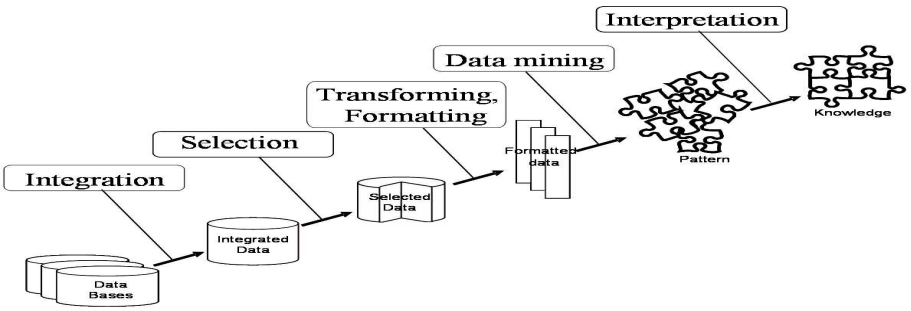


Figure 2: Le processus d'ECBD

Remarque 1 Supposons qu'on ait extrait la règle suivante d'une base de données :

$$\text{blanc} \Rightarrow \text{né aux États-Unis} \qquad \text{avec une probabilité de } 99,074\%$$

Cette règle est totalement dépendante des données étudiées : elle donne des connaissances *sur* la base de données (qui concerne un échantillon de recensement de la population du Kansas en 1990([1])). C'est le même problème que celui de l'échantillonnage en statistique.

1.2.3 Principes

Entrées et sorties du processus :

- Aller des données vers les connaissances (cf. section 1.1.1).
- Les données sont des structures *hétérogènes*, on les trouve par exemple dans des bases de données relationnelles ou objets, sous forme de liste de fichiers, sur des pages Web, dans des textes.
- Les connaissances nécessitent une syntaxe et une sémantique.

Étapes principales du processus d'ECBD :

1. Integration et séléction :
   Intégration des sources de données intéressantes et séléction des données intéressantes.
2. Transformations et formatage :
   données brutes $\mapsto$ données préparées et formatées
   - Exemples de transformation : enlever les données bruitées, traiter les données manquantes.
   - Formatage : formater les données sous une forme traitable par les algorithmes de fouille choisis. Exemples : définir des intérvalles de valeurs, binariser des données.
3. ***Fouille de données*** :
   données préparées et formatées $\mapsto$ éléments d'information extraits.
   C'est l'étape algorithmique du processus d'ECBD.
   Mise en oeuvre de *méthodes de fouilles*.
4. Interprétation :
   éléments d'information extraits $\mapsto$ unité de connaissances.

Le *processus d'ECBD* est un processus itératif et interactif :

[1] Cet exemple est repris de l'introduction de la thèse d'Yves Bastide.

- itératif : se fait en plusieurs passes ;
- interactif : l'*analyste* est "dans la boucle" et il oriente le processus selon ses *connaissances du domaine des données* et selon son intuition.
  - L'analyste est un expert du domaine des données (un astronome, un chimiste, un biologiste), s'il connait les principes de l'informatique et de l'ECBD, c'est un (grand) plus.
  - C'est généralement l'analyste qui effectue l'*interprétation*.
  - Les éléments d'information extraits des processus de fouille doivent être sous une forme compréhensible par l'analyste (intelligibilité).
  - Il est utile d'avoir des connaissances du domaine représentées et manipulables informatiquement pour assister l'analyste (exemple : si une connaissance extraite n'est *pas nouvelle* et qu'elle est représentée en machine, inutile de demander une confirmation de l'analyste).
    Ces connaissances du domaine sont souvent sous la forme d'une (ou plusieurs) ontologie(s).

1.2.4 Objectifs

- Caractérisation, discrimination
  Caractérisation : identification de critères d'appartenance à une classe,
  Discrimination : identification de critères discriminant entre l'appartenance à deux classes.
- Extraction de règles d'association
  Recherche de relations "Attribut-Valeur" qui occurent ensemble fréquemment.
- Classification
  Construction d'un modèle (ou d'une fonction) à partir du jeu de données, pour permettre la prédiction de l'appartenance d'un nouvel objet à une classe.
- Clustering
  Regroupement d'objets de sorte à minimiser la distance entre objets *similaires* et à maximimser la distance entre objets *différents*.
- Analyse de cas extrêmes

1.2.5 Applications

Exemples de bases de données étudiées dans le cadre de l'ECBD :

- Transactions de super-marchés ;
- Enregistrements de banques, d'assurances ;
- Immatriculations de voitures ;
- Données météorologiques ;
- Données en astronomie ;
- Réactions en chimie organique ;
- Données médicales ;
- Génomes.

1.2.6 Limites

Deux *points critiques* au niveau des données :

- À cause du volume de données étudiées, il faut faire un *passage à l'échelle* pour passer
  - des algorithmes d'apprentissage standards (qui ne sont applicables, en pratique, que sur des volumes de données relativement faibles) ;
  - aux algorithmes qui prennent en compte de très grands volumes de données (généralement stockées sur disque : le temps d'accès à ces données est nettement plus long que si elles étaient en mémoire centrale).
- Qualité des données :
  - Problèmes dus à l'absence de données (il en manque) et aux bruits (données erronées) ;
  - Nécessite d'avoir des méthodes pour améliorer cette qualité (notamment : filtrage).

2 Les méthodes de fouille de données : généralités

Il existe plusieurs moyens de différencier les méthodes de fouille de données. Une moyen classique est de distinguer les méthodes selon leur objectif (cf. section : 1.2.4) :

- les méthodes descriptives
  - Caractérisation, discrimination
  - Extraction de règles d'association
  - Analyse de cas extrême
- Les méthodes prédictives
  - Classification
  - Clustering

Classiquement, on distingue aussi les méthodes dites supervisées des non-supervisées. Nous préférerons, dans ce cours, une classification fonction du type de données qu'elles traitent en entrées. Nous distinguons alors 3 catégories :

1. Les méthodes symboliques, notamment :
   - L'extraction de motifs fréquents (cf. section 3) ;
   - L'extraction de règles d'association (cf. section 4) ;
   - La classification par treillis (cf. section 5) ;
   - La classification par arbres de décision.
2. Les méthodes numériques (issues, pour la plupart, du domaine de la reconnaissance des formes), notamment :
   - Les méthodes statistiques ;
   - Les méthodes d'analyse de données (classification automatique, classification par composantes principales) ;
   - Les modèles de Markov caché d'ordre 1 et 2 (donne notamment de très bons résultats en fouille de données génomiques) ;
   - Les méthodes neuronales ;
   - Les algorithmes génétiques.
3. Cas particulier de la fouille de textes : par exemple, fouille de base de données bibliographiques d'un domaine pour décrire ce domaine, en extrayant des textes des règles et des faits.

3 L'extraction de motifs fréquents

3.1 Motivation

L'extraction de motifs fréquents est une technique très utilisée en fouille de données et s'appuyant sur des principes relativement simples. Son objectif est de trouver les "motifs" qui apparaissent fréquemment dans une base de données. Un exemple très connu est le suivant. On dispose d'une base de données des tickets de caisse d'un supermarché. L'objectif est d'analyser les achats qui sont fréquemment associés, autrement dit, les motifs fréquents dans les tickets de caisse. On a trouvé ainsi que les couches pour nourrissons sont souvent associés à la bière...

3.2 Principe

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 (indiquant la présence ou l'absence d'une propriété) :

Définition 1 (base de données formelle) *Une base de données formelle est la donnée d'un triplet $(\mathcal{O}, \mathcal{P}, \mathcal{R})$ où :*

- *$\mathcal{O}$ est un ensemble fini d'objets ;*
- *$\mathcal{P}$ est un ensemble fini de propriétés ;*
- *$\mathcal{R}$ est une relation sur $\mathcal{O} \times \mathcal{P}$ qui permet d'indiquer si un objet x a une propriété p (noté $x\ \mathcal{R}\ p$).*

Considérons par exemple la base de données formelle suivante (qui va être utilisée tout au long du cours) :

$\mathcal{R}$	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

Où

- $\mathcal{O} = \{x_1, x_2, x_3, x_4, x_5, x_6\}$,
- $\mathcal{P} = \{a, b, c, d, e\}$,
- $x\ \mathcal{R}\ p$ si et seulement si la ligne de x et la colonne de p se croisent sur un 1 (et pas sur un 0), par exemple : $x_1\ \mathcal{R}\ a$, $x_1\ \mathcal{R}\ c$ et $x_1\ \mathcal{R}\ d$.

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

Pour la base de données en exemple, on a donc :

1 **motif de taille** 0 **:** $\emptyset$.

5 **motifs de taille** 1 **:** $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$ et $\{e\}$, qu'on notera, pour simplifier, $\underline{a}$, $\underline{b}$, $\underline{c}$, $\underline{d}$ et $\underline{e}$.

10 **motifs de taille** 2 **:** $\underline{ab}$, $\underline{ac}$, $\underline{ad}$, $\underline{ae}$, $\underline{bc}$, $\underline{bd}$, $\underline{be}$, $\underline{cd}$, $\underline{ce}$ et $\underline{de}$.

10 **motifs de taille** 3 **:** $\underline{abc}$, $\underline{abd}$, $\underline{abe}$, $\underline{acd}$, $\underline{ace}$, $\underline{ade}$, $\underline{bcd}$, $\underline{bce}$, $\underline{bde}$, et $\underline{cde}$.

5 **motifs de taille** 4 **:** $\underline{abcd}$, $\underline{abce}$, $\underline{abde}$, $\underline{acde}$ et $\underline{bcde}$.

1 **motif de taille** 5 **:** $\underline{abcde}$.

Dans cette base, x_1 possède les motifs $\emptyset$, $\underline{a}$, $\underline{c}$, $\underline{d}$, $\underline{ac}$, $\underline{ad}$, $\underline{cd}$ et $\underline{acd}$.

Parmi cet ensemble de $|2^{\mathcal{P}}| = 2^{|\mathcal{P}|}$ motifs, on va chercher ceux qui apparaissent fréquemment. Pour cela, on va introduire les notions de connexion de Galois et de support d'un motif :

Définition 3 (connexion de Galois (f, g)) *La connexion de Galois associée à une base de données formelle $(\mathcal{O}, \mathcal{P}, \mathcal{R})$ est le couple de fonctions (f, g) définies par :*

$$\begin{aligned} f:\ & 2^{\mathcal{P}} \longrightarrow 2^{\mathcal{O}} \\ & m \longmapsto f(m) = \{x \in \mathcal{O} \mid x \text{ possède } m\} \\ g:\ & 2^{\mathcal{O}} \longrightarrow 2^{\mathcal{P}} \\ & X \longmapsto g(X) = \{p \in \mathcal{P} \mid \forall x \in X,\ x\ \mathcal{R}\ p\} \end{aligned}$$

*g est dit dual de f et f, dual de g. On dit parfois que $f(m)$ est l'*image *du motif m.*

Remarque 2 Pour cette section, seule f est utile. g sera utilisé plus loin (section 5).

Définition 4 (support d'un motif) *Soit $m \in 2^{\mathcal{P}}$, un motif. Le support de m est la proportion d'objets dans $\mathcal{O}$ qui possèdent le motif :*

$$\begin{aligned} \texttt{support}:\ & 2^{\mathcal{P}} \longrightarrow [0;1] \\ & m \longmapsto \texttt{support}(m) = \frac{|f(m)|}{|\mathcal{O}|} \end{aligned}$$

Exemples, sur la base de données donnée en exemple :

$$\texttt{support}(\underline{a}) = \frac{3}{6} \qquad \texttt{support}(\underline{b}) = \frac{5}{6} \qquad \texttt{support}(\underline{ab}) = \frac{2}{6} \qquad \texttt{support}(\emptyset) = 1 \qquad \texttt{support}(\mathcal{P}) = 0$$

Propriété fondamentale : Le support est décroissant de $(2^{\mathcal{P}}, \subseteq)$ dans $([0;1], \leqslant)$.

Autrement dit, si m est un sous-motif de m' ($m \subseteq m'$) alors $\texttt{support}(m) \geqslant \texttt{support}(m')$.

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 σ_s :

Définition 5 (motif fréquent) *Soit $\sigma_s \in [0;1]$. Un motif m est fréquent (sous-entendu, relativement au seuil σ_s) si $\texttt{support}(m) \geqslant \sigma_s$. Sinon, il est dit non fréquent.*

Exercice 1 *On suppose qu'on a une base de données concernant une population d'humains adultes, constituée d'une relation dont les colonnes sont :*

- *numéro de sécurité sociale ;*
- *sexe ;*
- *âge ;*
- *taille ;*
- *couleur de cheveux.*

Ces colonnes ne sont pas à valeurs booléennes. On veut trouver les motifs fréquents dans cette base, mais pour cela, il faut la mettre sous la forme d'une base de données formelle $(\mathcal{O}, \mathcal{P}, \mathcal{R})$. Proposer et discuter une telle transformation.

3.3 Algorithme d'extraction des motifs fréquents

Une *approche naïve* pour l'extraction des motifs fréquents consiste à parcourir l'ensemble $2^{\mathcal{P}}$ de tous les motifs, à calculer leurs supports et à ne garder que les plus fréquents. Malheureusement, cette approche est trop consommatrice en temps, en effet, le nombre de motifs est $2^{|\mathcal{P}|}$ 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*.

Cette approche effectue une *extraction par niveaux*, 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.

Cette approche s'appuie sur les deux principes fondamentaux suivants (qui reposent sur la décroissance du support) :

- Tout sous-motif d'un motif fréquent est fréquent.
  (Si $m' \subseteq m$ et $\texttt{support}(m) \geqslant \sigma_s$, alors $\texttt{support}(m') \geqslant \sigma_s$).
- Tout super-motif d'un motif non fréquent est non fréquent.
  (Si $m' \supseteq m$ et $\texttt{support}(m) < \sigma_s$, alors $\texttt{support}(m') < \sigma_s$).

Le schéma d'algorithme suivant, baptisé `Apriori`, décrit l'extraction de motifs fréquents selon ce principe :

`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 σ_s.
Début
  $i \leftarrow 1$
  $\texttt{C}_1 \leftarrow$ ensemble des motifs de taille 1
  tant que $\texttt{C}_i \neq \emptyset$ **faire**
    Calculer le support de chaque motif $m \in \texttt{C}_i$
    $\texttt{F}_i \leftarrow \{m \in \texttt{C}_i \mid \texttt{support}(m) \geqslant \sigma_s\}$
    $\texttt{C}_{i+1} \leftarrow \texttt{générer-candidats}(\texttt{F}_i)$
    $i \leftarrow i + 1$
  fin-tant que
  retourner $\bigcup_{i \geqslant 1} \texttt{F}_i$
Fin

Sur notre exemple, le déroulement de l'algorithme se fait comme suit pour $\sigma_s = \frac{2}{6}$:

Génération de candidats de taille 1 :

$\texttt{C}_1$	$=$	$\{\underline{a},$	$\underline{b},$	$\underline{c},$	$\underline{d},$	$\underline{e}\}$
supports :		$\frac{3}{6}$	$\frac{5}{6}$	$\frac{5}{6}$	$\frac{1}{6}$	$\frac{5}{6}$

D'où $\texttt{F}_1 = \{\underline{a}, \underline{b}, \underline{c}, \underline{e}\}$ (aucun motif fréquent ne contiendra d).

Génération de candidats de taille 2 : Combiner 2 à 2 les candidats de taille 1 de $\texttt{F}_1$:

$\texttt{C}_2$	$=$	$\{\underline{ab},$	$\underline{ac},$	$\underline{ae},$	$\underline{bc},$	$\underline{be},$	$\underline{ce}\}$
supports :		$\frac{2}{6}$	$\frac{3}{6}$	$\frac{2}{6}$	$\frac{4}{6}$	$\frac{5}{6}$	$\frac{4}{6}$

$\texttt{F}_2 = \texttt{C}_2$: tous les motifs de $\texttt{C}_2$ sont fréquents.

Génération de candidats de taille 3 **:** Combiner 2 à 2 les candidats de taille 2 de $\mathtt{F}_2$ (et ne considérer que ceux qui donnent des motifs de taille 3) :

$$\begin{array}{lcl} \mathtt{C}_3 & = & \{\underline{abc}, \quad \underline{abe}, \quad \underline{ace}, \quad \underline{bce}\} \\ \text{supports :} & & \frac{2}{6} \qquad \frac{2}{6} \qquad \frac{2}{6} \qquad \frac{4}{6} \end{array}$$

$\mathtt{F}_3 = \mathtt{C}_3$: tous les motifs de $\mathtt{C}_3$ sont fréquents.

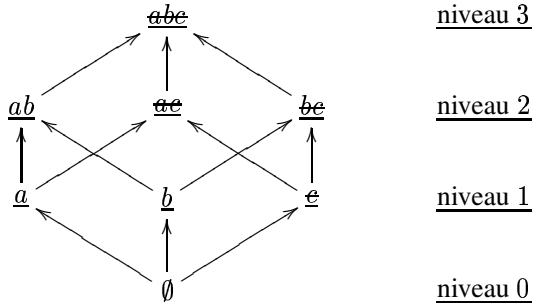
Génération de candidats de taille 4 **:**

$$\begin{array}{lcl} \mathtt{C}_4 & = & \{\underline{abce}\} \\ \text{support :} & & \frac{2}{6} \end{array}$$

Génération de candidats de taille 5 **:** $\mathtt{C}_5 = \emptyset$. Donc, $\mathtt{F}_5 = \emptyset$.

L'algorithme retourne alors l'ensemble des motifs fréquents $\mathtt{F}_1 \cup \mathtt{F}_2 \cup \mathtt{F}_3 \cup \mathtt{F}_4$.

Remarque 3 On peut voir cet algorithme comme le parcours du treillis des parties de $\mathcal{P}$ ordonné pour l'inclusion. Supposons par exemple que $\mathcal{P} = \{a, b, c\}$. Le treillis des parties de $\mathcal{P}$ est :



Il est parcouru par niveau croissant à partir du niveau $i = 1$. Quand un motif n'est pas fréquent, tous ses super-motifs sont non fréquents. Par exemple, dans cet exemple $\underline{c}$ n'est pas fréquent (il a été barré) et, par conséquent, aucun de ses super-motifs n'est considéré. On a ainsi élagué le parcours du treillis.

Remarque 4 Un des objectifs des optimisations de cet algorithme est de diminuer le nombre d'accès à la base de données.

Remarque 5 Le seuil σ_s est fixé par l'analyste. Celui-ci peut suivre une approche itérative en fixant un seuil au départ et, en fonction du résultat, changera la valeur du seuil :

- Si *trop* de motifs fréquents ont été trouvés, il augmentera le seuil ;
- Dans le cas inverse, il le diminuera.

Par ailleurs, on peut constater que le temps de calcul de l'algorithme `Apriori` décroit avec le seuil. Par conséquent, si l'analyste fixe une valeur de seuil trop grande, cela gaspillera moins de temps que s'il en fixe une trop petite.

(Merci à Hacène Cherfi pour une discussion à ce sujet.)

4 L'extraction de règles d'association

4.1 Motivation

Les règles d'association sont de la forme $R = p_1 \longrightarrow p_2$ 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.

4.2 Principe

On s'appuie également sur la notion de base de données formelle $(\mathcal{O}, \mathcal{P}, \mathcal{R})$.

Définition 6 (règle d'association) *Une règle d'association R est la donnée de deux motifs A et B. On la note $R = A \longrightarrow B$.*

Définition 7 (support et confiance d'une règle d'association)
Le support d'une règle $A \longrightarrow B$ est $\mathit{support}(A \longrightarrow B) = \mathit{support}(A \cup B)$.
La confiance d'une règle $A \longrightarrow B$ est $\mathit{confiance}(A \longrightarrow B) = \dfrac{\mathit{support}(A \cup B)}{\mathit{support}(A)}$.

Notons que $\texttt{confiance}(A \longrightarrow B) \in [0; 1]$ (cf. décroissance de `support`). La confiance d'une règle indique la proportion des objets qui possèdent à la fois A et B parmi les objets qui possèdent A :

$$\texttt{confiance}(A \longrightarrow B) = \frac{|f(A \cup B)|}{|f(A)|} = \frac{|f(A) \cap f(B)|}{|f(A)|}$$

$\texttt{confiance}(A \longrightarrow B) = 1$ signifie que tous les objets possédant le motif A possèdent le motif B.
$\texttt{confiance}(A \longrightarrow B) = 0$ signifie qu'aucun objet possédant le motif A ne possède le motif B.

Remarque 6 On peut rapprocher le support d'une règle d'une probabilité. En effet, $\texttt{support}(A \longrightarrow B)$ est la probabilité que x possède le motif A *et* que x possède le motif B (en supposant que x soit une variable aléatoire sur $\mathcal{O}$ avec une distribution uniforme).

On peut aussi rapprocher la confiance des probabilités conditionnelles. En effet, $\texttt{confiance}(A \longrightarrow B)$ est la probabilité que x possède le motif B sachant que x possède le motif A.

Parmi les règles, on s'intéresse aux règles dites valides :

Définition 8 (règle valide) *Une règle d'association R est* valide *si elle vérifie :*

- $\mathit{support}(R) \geqslant \sigma_s$ *et*
- $\mathit{confiance}(R) \geqslant \sigma_c$.

où $\sigma_s, \sigma_c \in [0; 1]$ sont des valeurs seuils prédéfinies.

Dans l'exemple cité en section 3.1, on avait le motif fréquent $m = \{\texttt{bières}, \texttt{couches}\}$. On s'intéresse donc aux règles $\texttt{bières} \longrightarrow \texttt{couches}$ et $\texttt{couches} \longrightarrow \texttt{bières}$. Si $\texttt{support}(m)$ est trop faible (inférieur à σ_s), alors la coocurrence des bières et des couches sur les tickets de caisse est trop faible pour être intéressante. Sinon, si la confiance de $\texttt{couches} \longrightarrow \texttt{bières}$ est forte, cela signifiera que, "souvent, quand on achète des couches, on achète des bières" (note personnelle : je pense que la confiance de $\texttt{bières} \longrightarrow \texttt{couches}$ sera beaucoup plus faible).

Définition 9 (règle exacte, règle approximative) *Une règle d'association R est* exacte *si* $\mathit{confiance}(R) = 1$. *Dans le cas contraire, R est dite* approximative.

Si $R = A \longrightarrow B$ est exacte, alors $\texttt{support}(A \cup B) = \texttt{support}(A)$.
Exemple : $\texttt{support}(\underline{a}) = \texttt{support}(\underline{ac}) = \frac{3}{6}$. Donc $a \longrightarrow c$ est une règle exacte.

Remarque 7 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.

4.3 Algorithme de génération de règles valides

Remarque 8 Si on a par exemple la règle $ab \longrightarrow ac$, on peut noter qu'elle a même support et même confiance que la règle $ab \longrightarrow c$: "répéter" en partie droite d'une règle une propriété apparaissant en partie gauche est inutile. De façon générale, on peut supposer que quand on a une règle $A \longrightarrow B$, alors $A \cap B = \emptyset$.

Fort de cette remarque, on peut noter les règles sous la forme $R = p_1 \longrightarrow p_2 \backslash p_1$ avec $p_1 \subseteq p_2$ (rappelons que $E \backslash F = \{x \mid x \in E \text{ et } x \notin F\}$). Par exemple, pour $ab \longrightarrow c$, on aura $p_1 = \underline{ab}$ et $p_2 = \underline{abc}$.

Pour une telle règle R, on a : $\texttt{support}(R) = \texttt{support}(p_2)$ et $\texttt{confiance}(R) = \dfrac{\texttt{support}(p_2)}{\texttt{support}(p_1)}$.

Le principe de l'algorithme est le suivant. On s'intéresse aux règles valides R, donc telles que $\texttt{support}(R) = \texttt{support}(p_2) \geqslant \sigma_s$. *A fortiori*, $\texttt{support}(p_1) \geqslant \sigma_s$ (cf. décroissance du support et $p_1 \subseteq p_2$).

1. On considère les règles de la forme $p_1 \longrightarrow p_2 \backslash p_1$ 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.).

Déroulement sur l'exemple du calcul des règles. On fixe les seuils $\sigma_s = \frac{2}{6}$ et $\sigma_c = \frac{2}{5}$.

Motifs fréquents de longueur 1 **:** $\underline{a}$, $\underline{b}$, $\underline{c}$ et $\underline{e}$ (pas de règle d'association à partir de ces motifs : il faut un motif de longueur au moins égale à 2 pour cela).

Motifs fréquents de longueur 2 **:** $\underline{ab}$, $\underline{ac}$, $\underline{ae}$, $\underline{bc}$, $\underline{be}$ et $\underline{ce}$.

Pour chacun de ces motifs fréquents m de longueur 2, on regarde les règles $p_1 \longrightarrow p_2 \backslash p_1$ avec $p_2 = m$:

- pour $m = p_2 = \underline{ab}$, deux règles peuvent être engendrées : $a \longrightarrow b$ (pour $p_1 = \underline{a}$) et $b \longrightarrow a$ (pour $p_1 = \underline{b}$). Le support de ces deux règles est le même : c'est $\texttt{support}(p_2) = \frac{2}{6}$. Le calcul de la confiance donne $\texttt{confiance}(a \longrightarrow b) = \frac{2/6}{3/6} = \frac{2}{3} \geqslant \sigma_c$ et $\texttt{confiance}(b \longrightarrow a) = \frac{2/6}{5/6} = \frac{2}{5} \geqslant \sigma_c$. On a donc deux règles valides : $\boxed{a \longrightarrow b}$ et $\boxed{b \longrightarrow a}$.
- pour $m = p_2 = \underline{ac}$, dont le support est $\frac{3}{6}$, on a les règles $\boxed{a \longrightarrow c}$ (de confiance $\frac{3}{3}$: règle exacte) et $\boxed{c \longrightarrow a}$ (de confiance $\frac{3}{5}$).
- pour $m = p_2 = \underline{ae}$, dont le support est $\frac{2}{6}$, on a les règles $\boxed{a \longrightarrow e}$ (de confiance $\frac{2}{3}$) et $\boxed{e \longrightarrow a}$ (de confiance $\frac{2}{5}$).
- pour $m = p_2 = \underline{bc}$, dont le support est $\frac{4}{6}$, on a les règles $\boxed{b \longrightarrow c}$ (de confiance $\frac{4}{5}$) et $\boxed{c \longrightarrow b}$ (de confiance $\frac{4}{5}$).
- pour $m = p_2 = \underline{be}$, dont le support est $\frac{5}{6}$, on a les règles $\boxed{b \longrightarrow e}$ (de confiance $\frac{5}{5}$: règle exacte) et $\boxed{e \longrightarrow b}$ (de confiance $\frac{5}{5}$: règle exacte).
- pour $m = p_2 = \underline{ce}$, dont le support est $\frac{4}{6}$, on a les règles $\boxed{c \longrightarrow e}$ (de confiance $\frac{4}{5}$) et $\boxed{e \longrightarrow c}$ (de confiance $\frac{4}{5}$).

Motifs fréquents de longueur 3 **:** $\underline{abc}$, $\underline{abe}$, $\underline{ace}$ et $\underline{bce}$

Pour chacun de ces motifs fréquents m de longueur 3, on regarde les règles $p_1 \longrightarrow p_2 \backslash p_1$ avec $p_2 = m$:

- pour $m = p_2 = \underline{abc}$, dont le support est $\frac{2}{6}$, on a les règles :
  1. à conclusions de longueur 1 : $\boxed{ab \longrightarrow c}$ (de confiance $\frac{2}{2}$), $\boxed{ac \longrightarrow b}$ (de confiance $\frac{2}{3}$) et $\boxed{bc \longrightarrow a}$ (de confiance $\frac{2}{4}$).

2. à conclusions de longueur 2 (obtenues par union des conclusions de longueur 1) : $\boxed{a \longrightarrow bc}$ (de confiance $\frac{2}{3}$), $\boxed{b \longrightarrow ac}$ (de confiance $\frac{2}{5}$) et $\boxed{c \longrightarrow ab}$ (de confiance $\frac{2}{5}$).

- pour $m = p_2 = \underline{abe}$, dont le support est $\frac{2}{6}$, on a les règles :
  1. à conclusions de longueur 1 : $\boxed{ab \longrightarrow e}$ (de confiance $\frac{2}{2}$: règle exacte), $\boxed{ae \longrightarrow b}$ (de confiance $\frac{2}{2}$: règle exacte) et $\boxed{be \longrightarrow a}$ (de confiance $\frac{2}{5}$).
  2. à conclusions de longueur 2 (obtenues par union des conclusions de longueur 1) : $\boxed{a \longrightarrow be}$ (de confiance $\frac{2}{3}$), $\boxed{b \longrightarrow ae}$ (de confiance $\frac{2}{5}$) et $\boxed{e \longrightarrow ab}$ (de confiance $\frac{2}{5}$).
- etc.

Motif fréquent de longueur 4 **:** $\underline{abce}$.

Pour ce motif fréquent $m = p_2$ de longueur 4, on regarde les règles $p_1 \longrightarrow p_2 \backslash p_1$ avec $p_2 = m$. Son support est $\frac{2}{6}$. On a les règles suivantes :

1. à conclusions de longueur 1 : $\boxed{abc \longrightarrow e}$ (de confiance $\frac{2}{2}$: règle exacte), $\boxed{abe \longrightarrow c}$ (de confiance $\frac{2}{2}$: règle exacte), $\boxed{ace \longrightarrow b}$ (de confiance $\frac{2}{2}$: règle exacte) et $\boxed{bce \longrightarrow a}$ (de confiance $\frac{2}{4}$). Toutes les conclusions de longueur 1 donnent des règles valides.
2. à conclusions de longueur 2 (obtenues par union des conclusions de longueur 1) : $\boxed{ce \longrightarrow ab}$ (de confiance $\frac{2}{4}$), $\boxed{be \longrightarrow ac}$ (de confiance $\frac{2}{5}$), $\boxed{bc \longrightarrow ae}$ (de confiance $\frac{2}{4}$), $\boxed{ae \longrightarrow bc}$ (de confiance $\frac{2}{2}$: règle exacte), $\boxed{ac \longrightarrow be}$ (de confiance $\frac{2}{3}$), $\boxed{ab \longrightarrow ce}$ (de confiance $\frac{2}{2}$: règle exacte).
3. à conclusions de longueur 3 (obtenues par union des conclusions de longueur 2) : $\boxed{e \longrightarrow abc}$ (de confiance $\frac{2}{5}$), $\boxed{c \longrightarrow abe}$ (de confiance $\frac{2}{5}$), $\boxed{b \longrightarrow ace}$ (de confiance $\frac{2}{5}$) et $\boxed{a \longrightarrow bce}$ (de confiance $\frac{2}{3}$).

L'exemple montre que cela fait beaucoup de règles (problème d'intelligibilité pour l'analyste).

Une stratégie d'élagage consiste à définir de nouvelles mesures de plausibilité (autres que le support et la confiance) pour limiter cet ensemble de règles.

Une deuxième stratégie consiste à élaguer les règles déjà contenues dans la base de connaissances. Si, par exemple, on dispose de l'ontologie de la figure 1, et qu'on a extrait la règle `tortue` $\longrightarrow$ `quadrupède, reptile`, cette règle peut être élaguée.

Une troisième stratégie consiste à constater que certaines règles sont moins informatives que d'autres. Par exemple, si on a les règles $R_1 = ab \longrightarrow c$ et $R_2 = a \longrightarrow cd$ avec le même support et la même confiance, alors, on pourra élaguer R_1, qui est une conséquence de R_2.

La remarque ci-dessous généralise cette troisième stratégie.

Remarque 9 Pour un motif donné, les règles valides les plus intéressantes sont à conclusions maximales (autrement dit : à prémisses minimales). En effet, si on a une règle $R = p_1 \longrightarrow p_2 \backslash p_1$ valide et p_1' tel que $p_1 \subseteq p_1' \subseteq p_2$, alors, avec $R' = p_1' \longrightarrow p_2 \backslash p_1'$, on a :

$$\texttt{support}(R') = \texttt{support}(R) = \texttt{support}(p_2)$$

$$\texttt{confiance}(R') = \frac{\texttt{support}(p_2)}{\texttt{support}(p_1')} \geqslant \frac{\texttt{support}(p_2)}{\texttt{support}(p_1)} = \texttt{support}(R)$$

Donc, si R est valide, alors R' l'est aussi.

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

5 La classification par treillis de Galois

5.1 Rappels sur les ordres et les treillis

Définition 10 (relation d'ordre) *Soit E un ensemble et $\leqslant$, une relation binaire sur E. $\leqslant$ est une* relation d'ordre *sur E si elle est réflexive ($\forall x \in E, x \leqslant x$), antisymétrique ($\forall x, y \in E, x \leqslant y$ **et** $y \leqslant x \Rightarrow x = y$) et transitive ($\forall x, y, z \in E, x \leqslant y$ **et** $y \leqslant z \Rightarrow x \leqslant z$). Dans ce cas, $(E, \leqslant)$ est un* ensemble ordonné.

Exemples d'ensemble ordonnés : $(\mathbb{R}, \leqslant)$, $(2^E, \subseteq)$, $(\mathbb{N}$, "divise"), (ensemble de mots, ordre lexicographique).

Définition 11 (diagramme de Hasse) *Un ensemble ordonné* fini $(E, \leqslant)$ *peut être représenté par un graphe dont les sommets sont les éléments de E et les arcs sont tels que :*

- *Pour tout x et tout y de E, $x \leqslant y$ si et seulement si il existe un chemin de x à y dans ce graphe ;*
- *Ce graphe ne contient pas de boucle (x, x) (on sait que $x \leqslant x$, car $\leqslant$ est réflexif) ;*
- *Ce graphe ne contient pas d'arc de transitivité (un arc de transitivité (x, y) est un arc tel qu'il existe un chemin de longueur supérieure ou égale à 2 qui relie x à y).*

Ce graphe est appelé diagramme de Hasse associé à $(E, \leqslant)$.

La figure 1 (page 3) peut être vue comme un diagramme de Hasse sur les classes d'animaux, la relation d'ordre sous-jacente étant l'inclusion.

Définition 12 (minorant, majorant, minimum, maximum) *Soit $(E, \leqslant)$ un ensemble ordonné. Soit $F \subseteq E$.*

- *Un* minorant *x de F est un élément de E tel que $\forall y \in F, x \leqslant y$.*
- *Un* majorant *x de F est un élément de E tel que $\forall y \in F, y \leqslant x$.*
- *Si x est un minorant de F et que $x \in F$, alors x est le* minimum *de F.*
- *Si x est un majorant de F et que $x \in F$, alors x est le* maximum *de F.*
- *On considère l'ensemble $\mathtt{m}$ des minorants de F. Si $\mathtt{m}$ a un et un seul maximum, ce maximum est appelé* borne inférieure *de F.*
- *On considère l'ensemble $\mathtt{M}$ des majorants de F. Si $\mathtt{m}$ a un et un seul minimum, ce minimum est appelé* borne supérieure *de F.*

Définition 13 (treillis) *Soit $(E, \leqslant)$ un ensemble ordonné. $(E, \leqslant)$ est un treillis si pour toute paire $\{x, y\}$ d'éléments de E, $\{x, y\}$ possède une borne inférieure (notée $x \wedge y$) et une borne supérieure (notée $x \vee y$).*

Exemples de treillis :

- $(2^E, \subseteq)$ avec $\wedge = \cap$ et $\vee = \cup$;
- $(\mathbb{N}$, "divise") avec $\wedge = \mathtt{pgcd}$ et $\vee = \mathtt{ppcm}$.

5.2 Motifs fermés et rectangles maximaux

Définition 14 (rectangle) *Un* rectangle *d'une base de données formelle $(\mathcal{O}, \mathcal{P}, \mathcal{R})$ est un couple (X, m) où $X \in 2^{\mathcal{O}}$, $m \in 2^{\mathcal{P}}$ et tel que pour tout $x \in X$, x possède le motif m.*

Sur l'exemple de la base de données formelle définie à la section 3.2, on a les rectangles : $(\{x_2, x_3\}, \underline{bc})$, $(\{x_1, x_3, x_5\}, \underline{a})$ et $(\{x_2, x_4\}, \underline{be})$.

Définition 15 (rectangle maximal) *Un* rectangle maximal *est un rectangle (X, m) tel que ni X ni m ne peuvent être étendus. Autrement dit, si on a un rectangle (X', m') tel que $X \subseteq X'$ et $m \subseteq m'$, alors $X' = X$ et $m' = m$.*

Remarque 10 Si (X, m) est un rectangle maximal, alors $f(m) = X$ et donc, $\texttt{support}(m) = \frac{|X|}{|\mathcal{O}|}$. En fait, si (X, m) est un rectangle maximal, alors m est un *fermé*, ce qui signifie que parmi les motifs m' tels que $f(m') = X$, m est le maximum unique. Les définitions suivantes formalisent cela.

Définition 16 (relation d'équivalence entre motifs) *Soit θ, la relation d'équivalence entre motifs définie par $m\ \theta\ m'$ si $f(m) = f(m')$.*
Soit $\mathcal{C}\ell(m)$ la classe d'équivalence pour θ contenant m : $\mathcal{C}\ell(m) = \{m' \in 2^{\mathcal{P}} \mid m'\ \theta\ m\}$.

Rappelons que l'ensemble des classes d'équivalence sur un ensemble E est une partition de E.
On peut montrer que $\mathcal{C}\ell(m)$ a un maximum unique. D'où la définition :

Définition 17 (motif fermé, motif clef) *Un motif m est* fermé *si m est le maximum de $\mathcal{C}\ell(m)$. Un motif m est* clef *si m est un minimum de $\mathcal{C}\ell(m)$.*

Remarque 11 La notion de motif clef n'est pas utilisée dans le reste du cours, mais est utilisée dans certains algorithmes optimisant `Apriori`, tel que l'algorithme `Pascal`.

Remarque 12 m est un motif fermé ssi $(f(m), m)$ est un rectangle maximal. Or si (X, m) est maximal, alors $g(X) = m$. Par conséquent, si m est fermé, on a $g(f(m)) = m$. On peut montrer que la réciproque est vraie :

Proposition 1 (caractérisation des motifs fermés) *: Soit $h = g \circ f$ et $h' = f \circ g$. On a les équivalences :*

(i) *$m \in 2^{\mathcal{P}}$ est un fermé ssi $h(m) = m$;*

(ii) *$X \in 2^{\mathcal{O}}$ est tel qu'il existe un rectangle maximal (X, m) ssi $h'(X) = X$.*

Remarque 13 Certains auteurs définissent un fermé de cette façon-là. D'ailleurs ce terme est justifié par la proposition suivante.

Proposition 2 *h et h' sont des opérateurs de fermeture, c'est-à-dire que ce sont des fonctions*

- *croissantes (si $m_1 \subseteq m_2$ alors $h(m_1) \subseteq h(m_2)$ et si $X_1 \subseteq X_2$ alors $h'(X_1) \subseteq h'(X_2)$),*
- *extensives ($m \subseteq h(m)$ et $X \subseteq h'(X)$) et*
- *idempotentes ($h \circ h = h$ et $h' \circ h' = h'$).*

m tel que $h(m) = m$ est un fermé pour *h, X tel que $h'(X) = X$ est un* fermé pour *h'.*

Remarque 14 Si on considère un motif quelconque m, le fermé qui est le maximum de $\mathcal{C}\ell(m)$ est $h(m)$. Par conséquent, $f(h(m)) = f(m)$ et $\texttt{support}(h(m)) = \texttt{support}(m)$.
Par exemple, $h(\underline{a}) = g(f(\underline{a})) = g(\{x_1, x_3, x_5\}) = \underline{ac}$ et $\texttt{support}(\underline{a}) = \texttt{support}(\underline{ac})$.

5.3 Treillis de Galois d'une base de données formelle

La proposition suivante est essentielle pour la suite :

Proposition 3 (propriété des rectangles maximaux) *Soient (X, m) et (X', m'), deux rectangles maximaux. On a l'équivalence*

$$X \subseteq X' \qquad \Leftrightarrow \qquad m \supseteq m'$$

Soit $\mathcal{T}_\mathcal{O}$ l'ensemble des fermés pour h' et $\mathcal{T}_\mathcal{P}$ l'ensemble des fermés pour h (i.e., des motifs fermés). La proposition ci-dessus indique que les ensembles ordonnés $(\mathcal{T}_\mathcal{O}, \subseteq)$ et $(\mathcal{T}_\mathcal{P}, \supseteq)$ sont isomorphes. Soit $\mathcal{T} = \mathcal{T}_\mathcal{O} \times \mathcal{T}_\mathcal{P}$ et soit $\leqslant$ la relation sur $\mathcal{T}$ définie par $(X, m) \leqslant (X', m')$ si $X \subseteq X'$ (ou, de façon équivalente, si $m \supseteq m'$).

Proposition 4 ($(\mathcal{T}, \leqslant)$ est un treillis) *$\leqslant$ est un ordre sur $\mathcal{T}$. De plus, $(\mathcal{T}, \leqslant)$, $(\mathcal{T}_\mathcal{O}, \subseteq)$ et $(\mathcal{T}_\mathcal{P}, \supseteq)$ sont trois treillis et ils sont isomorphes deux à deux.*

$(\mathcal{T}, \leqslant)$ est appelé treillis de Galois associé à $(\mathcal{O}, \mathcal{P}, \mathcal{R})$.

5.4 Motivation pour la création d'un treillis de Galois

À partir d'une base de données formelle $(\mathcal{O}, \mathcal{P}, \mathcal{R})$, on suppose qu'on a construit un treillis de Galois $(\mathcal{T}, \leqslant)$ et sa représentation sous la forme d'un diagramme de Hasse. Nous allons voir l'utilité de cette représentation.

On va considérer le treillis de Galois de la figure 3. Ce treillis a été construit à partir de la base de données formelle définie à la section 3.2.

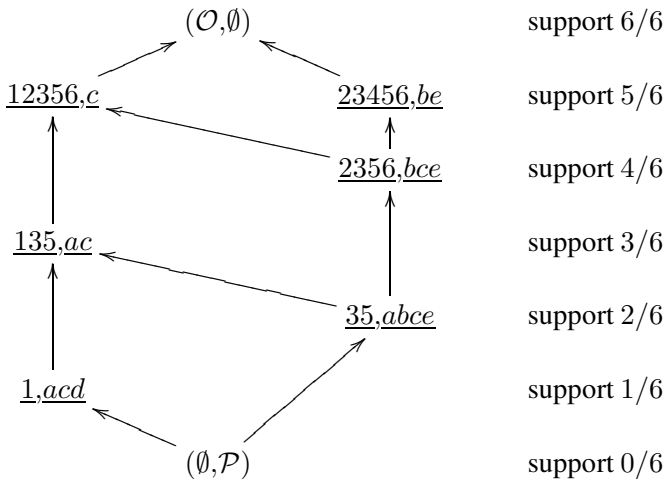


Figure 3: Un treillis de Galois (convention d'écriture : $\underline{135}, \underline{ac}$, par exemple, représente le rectangle $(\{x_1, x_3, x_5\}, \{a, c\})$).

Recherche de motifs fréquents. Le treillis comprend tous les motifs fermés. De plus, il est organisé par support croissant. Donc, pour obtenir tous les motifs fermés fréquents, il suffit de parcourir les sommets (X, m) du treillis à partir du maximum $(\mathcal{O}, \emptyset)$, à $|X|$ décroissant, jusqu'à ce que l'on ait $|X| < \sigma_s \cdot |\mathcal{O}|$. À partir des motifs fermés fréquents m, on peut obtenir tous les motifs fréquents : ce sont les sous-ensembles de m (rappelons que tout sous-motif d'un motif fréquent est fréquent).

De plus, on peut déterminer le support d'un motif m à partir du treillis.

Si ce motif est fermé, alors $(X, m) \in \mathcal{T}$, avec $X = f(m)$. Son support est alors $|X|/|\mathcal{O}|$. Par exemple, $\texttt{support}(\underline{abce}) = |\{x_3, x_5\}|/|\mathcal{O}| = 2/6$.

Si m *n'est pas* fermé, alors son support est le support du fermé $h(m)$ (cf. remarque 14). On utilise le fait que

$$\texttt{support}(m) = \max_{(X',m') \in \mathcal{T} \textbf{ et } m \subseteq m'} \texttt{support}(m')$$

En effet, comme $\texttt{support}(m) = \texttt{support}(h(m))$, il suffit d'identifier le rectangle maximal (X', m') tel que $m' = h(m)$ et on peut montrer que c'est celui qui maximisera le support. Par exemple, soit $m = \underline{a}$. Ce n'est pas un fermé (sinon, il apparaîtrait dans le treillis). Les motifs fermés qui le contiennent sont $\underline{ac}$, $\underline{acd}$, $\underline{abce}$ et $\mathcal{P}$. Or $\underline{ac} \subsetneq \underline{acd} \subsetneq \underline{abce} \subsetneq \mathcal{P}$. Donc, $\underline{ac}$ est inclus dans les autres ; c'est donc lui qui maximisera le support. Par conséquent, $\texttt{support}(\underline{a}) = \texttt{support}(\underline{ac}) = 3/6$.

Recherche de règles d'association. On peut également trouver des règles d'association dans un treillis de Galois. Pour cela, il y a (au moins) deux méthodes.

La ***première méthode*** permet de générer des règles exactes. Elle prend comme point de départ un rectangle maximal (X, m) de support supérieur ou égal au seuil σ_s. On s'intéresse aux règles $R = p_1 \longrightarrow p_2 \backslash p_1$ avec $m = p_2$ et $p_1 \subseteq p_2$. Rappelons que $\texttt{support}(R) = \texttt{support}(m)$). On choisit les p_1 dans $\mathcal{C}\ell(p_2)$ (pour cela, on procède de la même façon que ci-dessus, pour la recherche du support d'un motif non fermé). On a alors, $\texttt{confiance}(R) = \dfrac{\texttt{support}(p_2)}{\texttt{support}(p_1)} = 1$, puisque $p_1 \ \theta \ p_2$. Par exemple, le rectangle $\underline{35}, \underline{abce}$ permet de générer les règles exactes $ab \longrightarrow ce$ et $ae \longrightarrow bc$ (de support $2/6$).

Notons que parmi ces règles R, les plus informatives (au sens où elles "contiennent" les autres) sont celles pour lesquelles p_1 est un motif clef (mais comme nous n'avons pas parlé de la génération de motif clef...).

La ***deuxième méthode*** s'appuie sur deux rectangles (X_1, m_1) et (X_2, m_2) tels que $m_1 \subseteq m_2$. Ces rectangles permettent de générer les règles approximatives $m_1 \longrightarrow m_2 \backslash m_1$ de support $\texttt{support}(m_2) = |X_2|/|\mathcal{O}|$ et de confiance $\texttt{support}(m_2)/\texttt{support}(m_1) = |X_2|/|X_1|$.

Aide à la construction d'ontologies. Les rectangles maximaux sont aussi appelés *concepts formels* (ou *concepts*, pour faire plus court). Si $C = (X, m)$ est un concept, alors X est appelé *extension* de C et m est appelé *intension* (avec un "s") de C. Pour chaque concept du treillis de Galois, l'analyste peut choisir ou non de le réifier. S'il le réifie, il peut lui donner un nom (existant dans le domaine ou nouveau). On obtient de cette façon-là une proposition d'ontologie d'un domaine, pour le lien de généralité entre concepts. La qualité de cette ontologie dépendra de l'ensemble d'apprentissage $\mathcal{O}$ et de la qualité des données.

5.5 Construction d'un treillis de Galois

Une démarche simple pour construire un treillis de Galois s'appuie sur le principe suivant :

$$\text{Si } (X, m) \text{ et } (X', m') \text{ sont deux rectangles, alors } (X \cup X', m \cap m') \text{ est un rectangle.} \qquad (1)$$

Le principe consiste à travailler à taille de rectangle X croissante :

1. Initialisation par les rectangles de taille $X = 1$:
   pour tout $x \in \mathcal{O}$, on construit le rectangle $(\{x\}, g(\{x\}))$.

2. On construit les rectangles de taille $|X| = 2$ en combinant les rectangles de taille 1 grâce à la formule (1). On ne considère que les rectangles à motifs non vides.

   Pour chaque rectangle (X, m) de taille 2 ainsi généré, on élague les rectangles (X', m) (le *même* m) de taille 1 tels que $X' \subseteq X$.

3. On construit les rectangles de taille 3 à partir des rectangles de taille 2 dont on élague une partie.

4. On itère jusqu'à ce qu'aucun élément maximal ne puisse être trouvé.

5. On ajoute le rectangle $(\emptyset, \mathcal{P})$ s'il est maximal (i.e., s'il n'existe pas d'objet ayant toutes les propriétés) et le rectangle $(\mathcal{O}, \emptyset)$ s'il est maximal (i.e., s'il n'existe pas de propriété partagée par tous les objets) et on obtient tous les rectangles maximaux. On les organise en diagramme de Hasse pour l'ordre $\leqslant$ sur les treillis de Galois.

Nous allons voir le déroulement de cette construction, sur la base de données formelle définie à la section 3.2. Les conventions d'écritures sont celles de la figure 3. De plus, les rectangles encadrés sont ceux qui *ne sont pas* élagués à l'itération suivante.

Taille 1 : $\boxed{1, acd}$, $2, bce$, $3, abce$, $4, be$, $5, abce$ et $6, bce$.

Taille 2 : $12, c$, $13, ac$, $15, ac$, $16, c$, $23, bce$, $24, be$, $25, bce$, $26, bce$, $34, be$, $\boxed{35, abce}$, $36, bce$, $45, be$, $46, be$ et $56, bce$.

Taille 3 : $123, c$, $125, c$, $126, c$, $\boxed{135, ac}$, $136, c$, $156, c$, $234, be$, $235, bce$, $236, bce$, $245, be$, $246, be$, $256, bce$, $345, be$, $346, be$, $356, bce$ et $456, be$.

Taille 4 : $1235, c$, $1236, c$, $1256, c$, $1356, c$, $2345, be$, $2346, be$, $\boxed{2356, bce}$, $2456, be$ et $3456, be$.

Taille 5 : $\boxed{12356, c}$ et $\boxed{23456, be}$.

6 À propos d'outils de fouille de données

Weka est une boîte à outils d'apprentissage gratuite et *open source* (en Java) et téléchargeable à `http://www.cs.waikato.ac.nz/ml/weka/`. Elle contient des outils utiles à la fouille de données, notamment pour la recherche de motifs fréquents et de règles d'association.

Une boîte à outils Java contenant des outils efficaces de fouille de données (recherche de motifs fermés fréquents, de motifs rares, etc.) s'appelle Coron. Son créateur est Laszlo Szathmary de l'équipe Orpailleur du Loria. Il devrait être disponible sur la page `http://coron.loria.fr/`.

Bibliographie

[1] Han (J.), Kamber (M.). – Data Mining: Concepts and Techniques. *Morgan Kaufmann Publishers*, 2001.

[2] Bastide (Y.), Taouil (R.), Pasquier (N.), Stumme (G.) et Lakhal (L.). – PASCAL : un algorithme d'extraction des motifs fréquents. *Technique et Science Informatiques (TSI)*, vol. 21, n° 1, 2002, pp. 65–95.

[3] Godin (R.), Mineau (G.), Missaoui (R.) et Mili (H.). – Méthodes de classification conceptuelle basées sur les treillis de Galois et applications. *Revue d'Intelligence Artificielle*, vol. 9, n° 2, 1995, pp. 105–137.

[4] Stumme (G.), Taouil (R.), Bastide (Y.), Pasquier (N.) et Lakhal (L.). – Computing Iceberg Concept Lattices with Titanic. *Journal of Data and Knowledge Engineering*, vol. 42, n° 2, 2002, pp. 189–222.