Conception et implémentation de nouvelles fonctionnalités dans un prototype de fouille de données

Kévin Emamirad

Université du Québec en Outaouais
Département d'informatique et d'ingénierie

emak01@uqo.ca

mercredi 31 mai 2017

Plan de la présentation

Introduction

- L'analyse formelle de concepts (AFC) (Ganter et Wille 1999) est un formalisme de représentation des connaissances et de fouille de données qui est
  - utilisé dans divers domaines (informatique, linguistique, sociologie, biologie, etc.), et
  - produit des visualisations graphiques des structures inhérentes aux données sous forme de treillis de concepts (Galois).
- Au cours des deux dernières décennies, on a vu apparaître plusieurs d'outils d'analyse formelle de concepts tels *ToscanaJ, ConExp, Coron, Java Lattices*, et *Lattice Miner*.

Introduction

Objectifs

- But : Enrichir *Lattice Miner* (Roberge 2007), un prototype de fouille de données qui exploite l'AFC avec les objectifs suivants :
  1. production d'implications avec négation (Missaoui, Nourine et Renaud 2012),
  2. enrichissement du module de génération des règles triadiques pour obtenir exhaustivement et précisément les trois formes d'implications triadiques définies par (Ganter et Obiedkov 2004),
  3. validation intensive de la procédure de production de la base d'implications de (Guigues et Duquenne 1986) et de la procédure de calcul des relations de flèches. Cette dernière est utile dans le processus de décomposition de contextes formels (Viaud et al. 2015), et
  4. amélioration de la convivialité de l'interface usager.

Rappels

Rappels I

Analyse formelle de concepts

Définition 1 : Contexte formel

Soit $\mathbb{K} = (G, M, I)$ *un contexte formel* où G, M et I sont respectivement un ensemble d'objets, une collection d'attributs et une relation binaire entre G et M. L'expression $(g, m) \in I$ ou encore gIm signifie que l'objet g possède l'attribut m.

	a	b	c	d	e	f
1	×		×		×	
2		×		×		×
3			×	×		
4		×	×			×
5	×	×				×
6	×	×		×	×	

Tableau 1: Exemple de contexte $\mathbb{K}$

Rappels II

Analyse formelle de concepts

Définition 2 : Concept formel

Un *concept formel* c est une paire d'ensembles $c := (A, B)$ avec $A \subseteq G$, $B \subseteq M$, $A = B'$ et $B = A'$, où A' est l'ensemble des attributs partagés par les objets dans A et B' est l'ensemble des objets ayant tous leurs attributs dans B. Les sous-ensembles A et B sont appelés respectivement l'extension et l'intention du concept c. Les valeurs de A' et B' sont obtenues comme suit : $A' := \{m \in M \mid gIm\ \forall g \in A\}$ et $B' := \{g \in G \mid gIm\ \forall\ m \in B\}$.

Défintion 3 : Sous-ensemble fermé

Un sous-ensemble X est fermé si $X'' = X$. Un concept objet pour l'objet g est une paire de la forme $\gamma(g) = (g'', g')$ alors que le concept attribut pour l'attribut m est $\mu(m) = (m', m'')$. Les sous-ensembles fermés de G sont les extensions alors que les sous-ensembles fermés de M sont les intentions de $\mathbb{K}$.

Rappels III

Analyse formelle de concepts

Défintion 4 : Treillis de concepts

Un *treillis de concepts* $\underline{\mathfrak{B}}(G, M, I)$ est un treillis résultant de l'ordre partiel existant entre les concepts du contexte $\mathbb{K} = (G, M, I)$.

$$(A, B) \leq (C, D) \Leftrightarrow A \subseteq C \text{ et } D \subseteq B$$

(A, B) est alors un sous-concept ou prédecesseur immédiat de (C, D) alors que ce dernier est un successeur de (A, B).

Définition 5 : Les bornes du treillis

La borne inférieure $\wedge$ (*meet*) d'un ensemble de concepts (X_i, Y_i) avec $i = 1, \ldots k$ est le plus grand des prédécesseurs communs. De même, la borne supérieure $\vee$ (*join*) d'un ensemble de concepts est le petit des successeurs communs.

Rappels IV

Analyse formelle de concepts

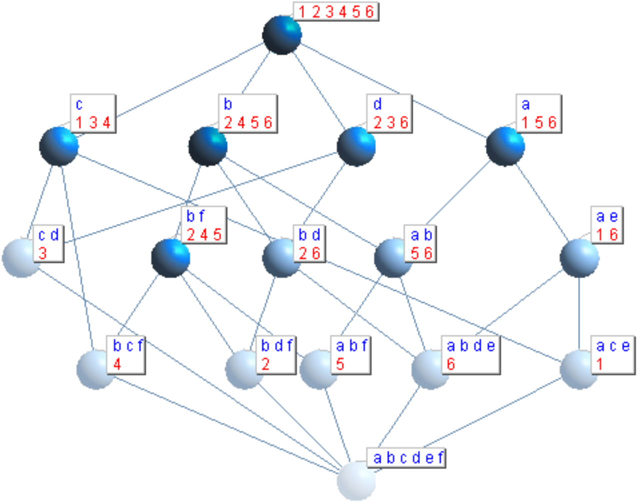


Figure 1: Treillis avec étiquetage complet pour le contexte du tableau 1

Rappels V

Analyse formelle de concepts

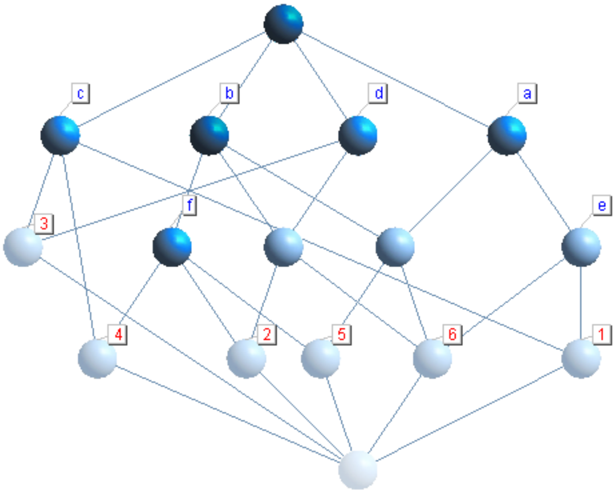


Figure 2: Treillis avec étiquetage réduit pour le contexte du tableau 1

Rappels VI

Analyse formelle de concepts

L'*infimum* est le plus petit concept d'un treillis de Galois alors que le *supremum* est son plus grand concept.
Lorsqu'un concept c_1 est plus petit qu'un autre concept c_2, alors le nœud représentant c_1 est placé plus bas que son successeur c_2. Lorsqu'une relation d'ordre existe entre les concepts c_1 et c_2, on dit alors que ces concepts sont *comparables* ($\leq$), sinon ils ne le sont pas ($\nleq$).

Définition 6 : sup-irréductible et inf-irréductible

On rappelle qu'un concept est dit *sup-irréductible* (ou *join-irréductible*) si et seulement si il possède un unique prédécesseur et il est dit *inf-irréductible* (ou *meet-irréductible*) s'il admet un unique successeur.

Rappels VII

Règles d'association et implications

Définition 7 : règle d'association

$$r : Y \rightarrow Z \ [sup, conf]$$

r est une règle d'association avec

- Y et Z sont des sous-ensembles d'attributs appelés *itemsets*,
- $Y \cap Z = \emptyset$,
- *sup*, le support est $Prob(Y \cup Z)$ (proportion des objets ayant simultanément les attributs Y et Z), et
- *conf*, la confiance est $Prob(Y \cup Z)$ (probabilité d'avoir Z lorsque Y est présent dans le contexte $\mathbb{K}$).

Rappels VIII

Règles d'association et implications

Définitions 8 : itemsets et générateurs

- Un *itemset* est dit *fréquent* si la proportion d'objets le possédant est au moins égale au support minimum défini par l'utilisateur.
- Un *itemset* Y est dit *fermé* si $Y = Y''$. Cela signifie que Y est une intention d'un concept formel.
- Un *générateur* G (Pfaltz et Taylor 2002) d'un itemset fermé (intention) Y est un ensemble minimal de Y tel que $G'' = Y$.

Rappels IX

Règles d'association et implications

Définition 9 : implication et base générique

$$r : g_i \rightarrow Int(c) \setminus g_i$$

r est une implication ou règle exacte avec

- c un concept formel,
- $G = \{g_1, \ldots g_i, \ldots g_n\}$ l'ensemble de ses générateurs,
- $Int(c)$ représente l'intention du concept c,
- $support(c)$ est la proportion d'objets contenus dans l'extension de c,
- $support(r) = support(c)$ et
- $confiance(r) = 100~\%$

Une base générique (Pasquier et al. 1999) d'implications est une représentation relativement concise d'implications de la forme précédente.

Rappels X

Règles d'association et implications

Définition 10 : règle approximative

$$r : g_i \rightarrow Int(c_j) \backslash Int(c)$$

r est une règle approximative avec

- c un concept formel,
- $P = \{c_1, \ldots, c_j, \ldots\}$ l'ensemble des prédecesseurs de c,
- $Int(c)$ représente l'intention du concept c,
- g_i un générateur de $Int(c)$,
- $support(r) = support(c_j)$, et
- $confiance(r) = support(c_j) / support(c)$

Rappels XI

Règles d'association et implications

Définitions 11 : pseudo-intent et base de Guigues-Duquenne

Un ensemble $P \subseteq M$ est un *pseudo-intent* de (G, M, I) (Ganter et Wille 1999 ; Guigues et Duquenne 1986) si et seulement si $P \neq P''$ et si $Q'' \subseteq P$ est vraie pour tout pseudo-intent $Q \subseteq P$, $Q \neq P$.
L'ensemble des implications de la forme :

$$\mathcal{L} := \{\mathcal{P} \rightarrow \mathcal{P}'' \setminus \mathcal{P} \mid \mathcal{P} \textit{ pseudo-intent } \}$$

est appelé base de Guigues-Duquenne (*stem base*) et reconnue comme étant minimale.

À titre d'exemple, $\{a, c\}$ est un pseudo-intent dont le fermé est $\{a, c, e\}$, d'où l'implication $\{a, c\} \rightarrow \{e\}$

Rappels XII

Règles d'association et implications

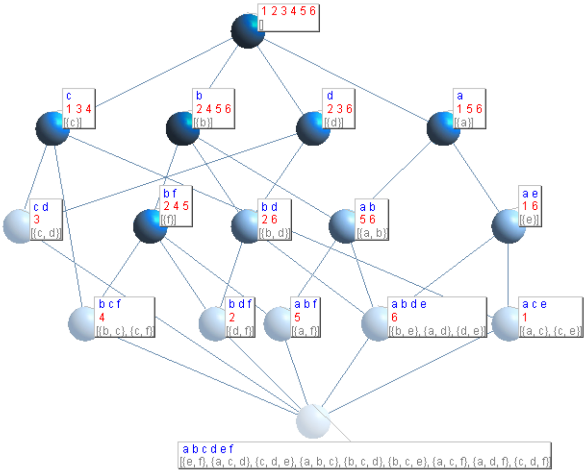


Figure 3: Treillis de concepts avec indication des générateurs

Rappels XIII

Règles d'association et implications

Définition 12 : contexte complémentaire $\tilde{\mathbb{K}}$

Soit un contexte $\mathbb{K} = (G, M, I)$ décrivant un ensemble G d'objets, un ensemble M de propriétés (attributs) et une relation binaire I entre G et M. Le contexte complémentaire de $\mathbb{K}$ est $\tilde{\mathbb{K}} = (G, \tilde{M}, G \times M \backslash I)$ avec $\tilde{M}$ l'ensemble des attributs négatifs.

Définition 13 : apposition du contexte $\mathbb{K}$ avec son complémentaire $\tilde{\mathbb{K}}$

Le contexte $\mathbb{K}|\tilde{\mathbb{K}}$ est l'apposition du contexte $\mathbb{K}$ avec son complémentaire $\tilde{\mathbb{K}}$. Cela signifie que $\mathbb{K}|\tilde{\mathbb{K}} := (G, M \cup \tilde{M}, I \cup G \times M \backslash I)$.

Rappels XIV

Règles d'association et implications

	a	b	c	d	e	f	$\tilde{a}$	$\tilde{b}$	$\tilde{c}$	$\tilde{d}$	$\tilde{e}$	$\tilde{f}$
1	×		×		×			×		×		×
2		×		×		×	×		×		×	
3			×	×			×	×			×	×
4		×	×			×	×			×	×	
5	×	×				×			×	×	×	
6	×	×		×	×				×			×

Figure 4: $\mathbb{K}|\tilde{\mathbb{K}}$: apposition du contexte $\mathbb{K}$ avec son complémentaire $\tilde{\mathbb{K}}$

Rappels XV

Relations flèches

Définition 14 : Relations flèches

La relation entre l'objet g et l'attribut m dans un contexte formel se présente sous l'une des quatre formes suivantes :

- $g \updownarrow m$ si $\gamma(g) \not\leq \mu(m)$, $\gamma(g) \leq m^+$, et $g^- \leq \mu(m)$
- $g \uparrow m$ si $\gamma(g) \not\leq \mu(m)$, $\gamma(g) \leq m^+$, et $g^- \not\leq \mu(m)$
- $g \downarrow m$ si $\gamma(g) \not\leq \mu(m)$, $\gamma(g) \not\leq m^+$, et $g^- \leq \mu(m)$
- $g \circ m$ si $\gamma(g) \not\leq \mu(m)$, $\gamma(g) \not\leq m^+$, et $g^- \not\leq \mu(m)$

où g^- représente le prédécesseur immédiat du concept objet (sup-irréductible) $\gamma(g)$ et m^+ représente le successeur immédiat du concept attribut (inf-irréductible) $\mu(m)$.

Rappels XVI

Relations flèches

	a	b	c	d	e	f
1	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\swarrow$
2	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\swarrow$	$\times$
3	$\swarrow\!\!\!\!\!\!\nearrow$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\times$	$\swarrow$	$\swarrow$
4	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\swarrow$	$\times$
5	$\times$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\swarrow\!\!\!\!\!\!\nearrow$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$
6	$\times$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$	$\times$	$\times$	$\swarrow\!\!\!\!\!\!\nearrow$

Figure 5: Relation flèches pour le contexte $\mathbb{K}$

Par exemple, $3 \downarrow e$ car

- $\gamma(3) \not\leq \mu(e)$ avec $\gamma(3) = (3, cd)$ et $\mu(e) = (16, ae)$
- $\gamma(3) \not\leq e^+$ avec $e^+ = (156, a)$
- $3^- \leq \mu(e)$ avec $3^- = (\emptyset, abcdef)$

Les outils de l'analyse formelle de concepts

Les outils de l'analyse formelle de concepts

Galicia

- Galicia [1] a été developpé en Java par Pekto Valchev et ses collaborateurs (Université de Montréal).

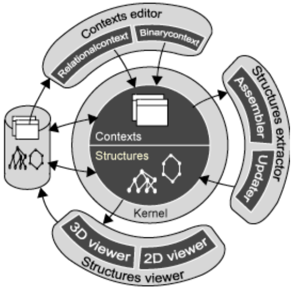


Figure 6: Cycle de vie du treillis Galicia

Quelques fonctionnalités de Galicia :

- Affichage du treillis en trois dimensions
- Manipulation du contexte et du treillis
- Plusieurs procédures de construction du treillis : Bordat, Nourine, Godin et *Next-Closure*
- Connection à des bases de données SQL pour sauvegarder les contextes

1. `http://www.iro.umontreal.ca/~galicia/`

Les outils de l'analyse formelle de concepts

Galicia

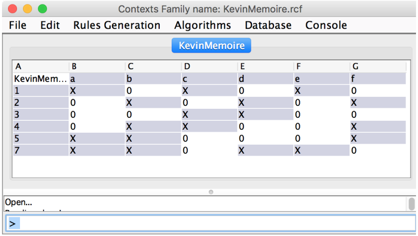


Figure 7: Éditeur de contexte

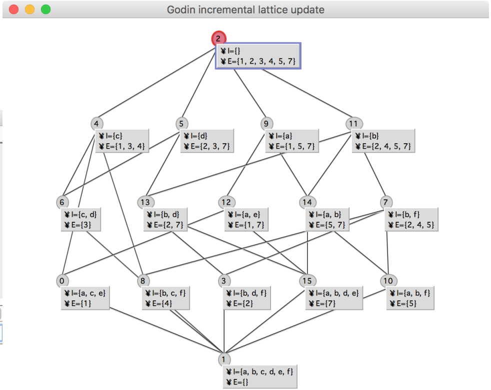


Figure 8: Treillis Galicia avec étiquetage complet

Les outils de l'analyse formelle de concepts

Le système Coron

Le système Coron [2] (Szathmary 2006) est une boîte à outils qui a été développée au laboratoire LORIA [3]

- Développé avec Java et Perl
- En ligne de commandes
- Version actuelle de ce système est 0.8 et date de janvier 2010
- Compatible avec les systèmes d'exploitation Unix, macOS et Windows
- Implémente plusieurs algorithmes de fouille de données tels qu'Apriori, Close, Aclose, Carpathia, Charm, Next-Closure ou NFI-Simple.
- Simple d'utilisation, modulaire et documentation riche
- Code source non disponible

2. http://coron.loria.fr/site/index.php.
3. Laboratoire lorrain de recherche en informatique et ses applications, Nancy, France

Les outils de l'analyse formelle de concepts

Librairie Java Lattices

La libraire *Java Lattices*[4] (Bertet et al. 2014) a été développée par le laboratoire L3i[5].

- Permet de générer des treillis de Galois et des règles d'association
- Code source libre et disponible sur la plate-forme GitHub
- Entièrement écrit avec Java
- La libraire est compatible avec Maven (outil Java pour gérer les dépendances)
- Client en ligne de commandes disponible
- Algorithme *Limited Object Access (LOA)* (Demko et Bertet 2011)

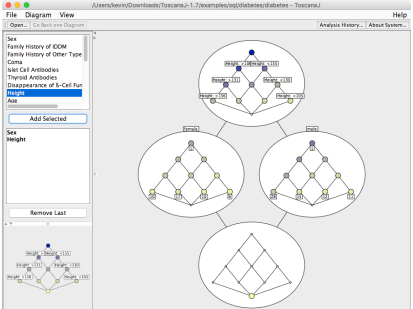
4. `https://thegalactic.github.io/java-lattices`
5. Laboratoire Informatique, Image et Interaction, université de La Rochelle, France

Les outils de l'analyse formelle de concepts

ToscanaJ

ToscanaJ (Becker, Hereth et Stumme 2002) est une réimplémentation avec le langage de programmation Java de l'outil d'AFC connu sous le nom de « *Toscana* ». Cet outil a la particularité d'être le premier outil à offrir l'affichage des diagrammes imbriqués (*nested line diagrams*) comme on peut le voir avec la figure 9.

Figure 9: ToscanaJ : Diagrammes imbriqués



Les outils de l'analyse formelle de concepts

Concept Explorer

Concept Explorer[6] (*ConExp*) est un outil souvent utilisé par les chercheurs en AFC.

- Interface graphique complète en Java Swing
- Édition et transformation de contextes
- Construction de treillis de concepts
- Code distribué librement
- Base Guigues-Duquenne
- Génération des règles d'association et l'exploration des attributs
- Génération des relations flèches

Dans le cadre de ce mémoire, nous allons utiliser *ConExp* à des fins de validation, notamment pour tester la procédure de production des relations flèches et pour la génération de la base de Guigues–Duquenne.

6. `http://conexp.sourceforge.net/`

Les outils de l'analyse formelle de concepts

Concept Explorer

```
1 < 1 > a b e =[100%]=> < 1 > d;
2 < 1 > a c =[100%]=> < 1 > e;
3 < 1 > a d =[100%]=> < 1 > b e;
4 < 1 > b c =[100%]=> < 1 > f;
5 < 2 > e =[100%]=> < 2 > a;
6 < 3 > f =[100%]=> < 3 > b;
7 < 4 > b =[75%]=> < 3 > f;
8 < 6 > { } =[67%]=> < 4 > b;
9 < 3 > d =[67%]=> < 2 > b;
10 < 3 > a =[67%]=> < 2 > e;
11 < 3 > a =[67%]=> < 2 > b;
12 < 6 > { } =[50%]=> < 3 > d;
13 < 6 > { } =[50%]=> < 3 > c;
14 < 6 > { } =[50%]=> < 3 > a;
15 < 2 > b d =[50%]=> < 1 > f;
16 < 2 > b d =[50%]=> < 1 > a e;
17 < 2 > a b =[50%]=> < 1 > f;
18 < 2 > a e =[50%]=> < 1 > c;
19 < 3 > d =[33%]=> < 1 > c;
20 < 3 > c =[33%]=> < 1 > b f;
```

Figure 10: *ConExp* : Règles d'association

`7 < 4 > b =[75%]=> < 3 > f;`

Il s'agit de la 7-ème règle d'association laquelle indique que *b* implique *f* avec un support absolu de 3 parmi les six objets (et donc un support relatif de 50%) et une confiance de 75%.

Les outils de l'analyse formelle de concepts

Concept Explorer

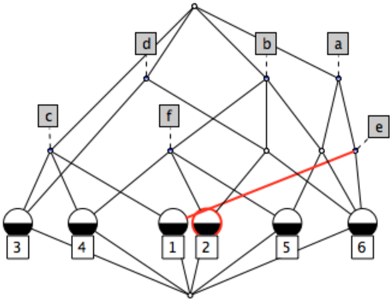


Figure 11: *ConExp* : Affichage du treillis

- L'affichage du treillis se fait selon un étiquetage réduit seulement.
- *ConExp* utilise un algorithme de construction du treillis avec le minimum d'intersections et les collisions sont affichées en rouge.

Les outils de l'analyse formelle de concepts

Lattice Miner

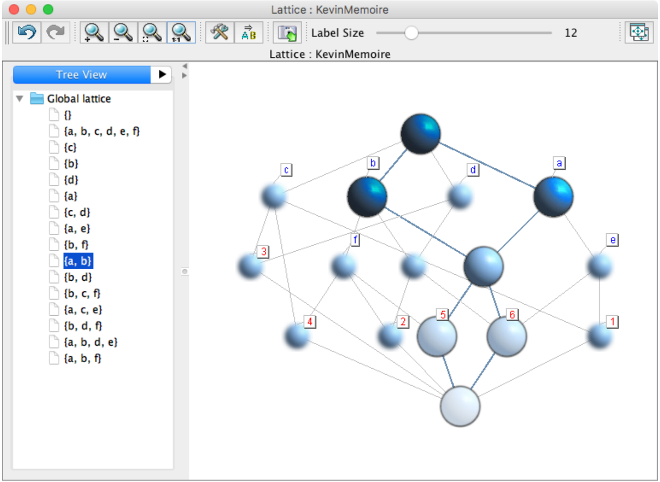
- Logiciel en Java pour la visualisation et la manipulation de treillis
- Intialement conçu par Geneviève Roberge dans le cadre de son mémoire de maîtrise.
- Continuellement enrichi par les membres de l'équipe et plusieurs stagiaires ingénieurs français.
- Aussi bien dans sa version initiale que dans sa version révisé, *Lattice Miner* est un logiciel libre disponible sur *SourceForge*[7] et GitHub[8].

7. `https://sourceforge.net/projects/lattice-miner/`
8. `https://github.com/LarimUQO/lattice-miner`

Les outils de l'analyse formelle de concepts

Lattice Miner

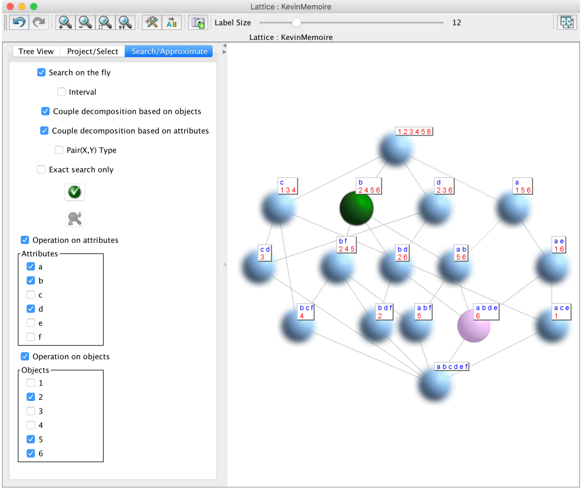
Figure 12: Génération du treillis de concepts dans *Lattice Miner*



Les outils de l'analyse formelle de concepts

Lattice Miner

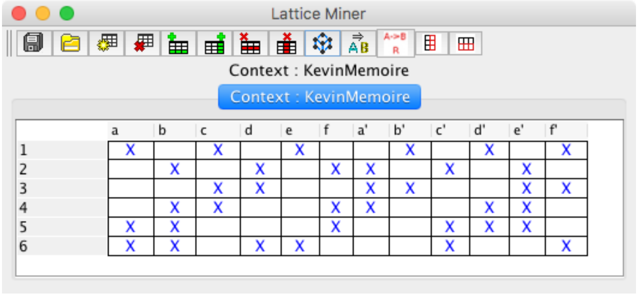
Figure 13: Approximation dans *Lattice Miner*



Les outils de l'analyse formelle de concepts

Lattice Miner

Figure 14: Affichage de l'apposition d'un contexte avec son complémentaire

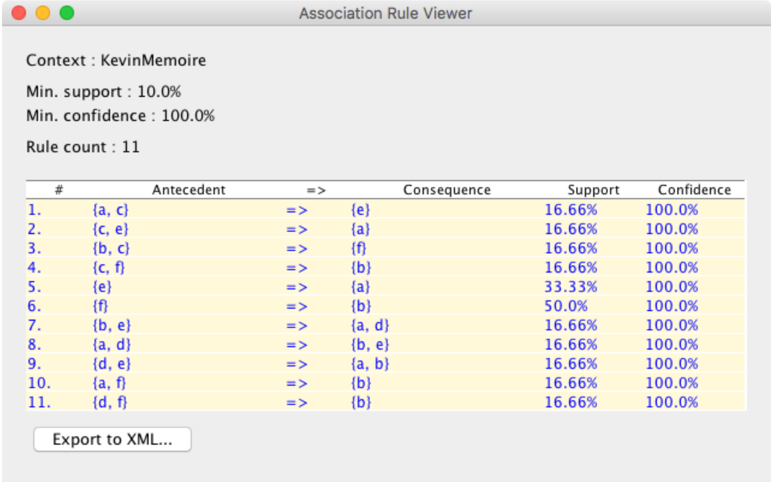


	a	b	c	d	e	f	a'	b'	c'	d'	e'	f'
1	X		X		X			X		X		X
2		X		X		X	X		X		X	
3			X	X			X	X			X	X
4		X	X			X	X			X	X	
5	X	X				X			X	X	X	
6	X	X		X	X				X			X

Les outils de l'analyse formelle de concepts

Lattice Miner

Figure 15: Base générique d'implications avec possibilité de redondance



Association Rule Viewer

Context : KevinMemoire

Min. support : 10.0%

Min. confidence : 100.0%

Rule count : 11

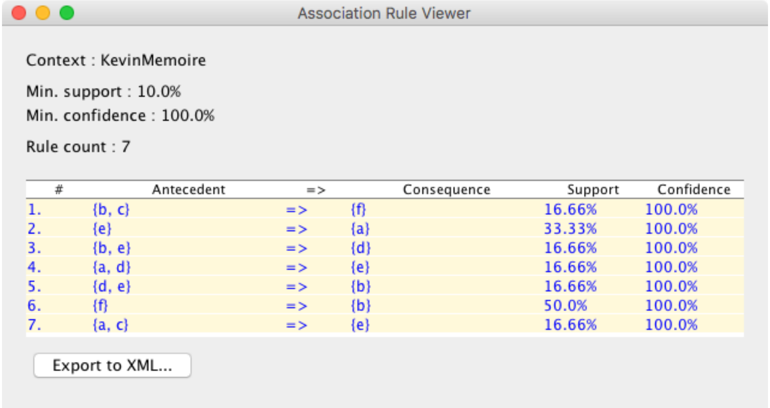
#	Antecedent	=>	Consequence	Support	Confidence
1.	{a, c}	=>	{e}	16.66%	100.0%
2.	{c, e}	=>	{a}	16.66%	100.0%
3.	{b, c}	=>	{f}	16.66%	100.0%
4.	{c, f}	=>	{b}	16.66%	100.0%
5.	{e}	=>	{a}	33.33%	100.0%
6.	{f}	=>	{b}	50.0%	100.0%
7.	{b, e}	=>	{a, d}	16.66%	100.0%
8.	{a, d}	=>	{b, e}	16.66%	100.0%
9.	{d, e}	=>	{a, b}	16.66%	100.0%
10.	{a, f}	=>	{b}	16.66%	100.0%
11.	{d, f}	=>	{b}	16.66%	100.0%

Export to XML...

Les outils de l'analyse formelle de concepts

Lattice Miner

Figure 16: Base générique d'implications sans redondance

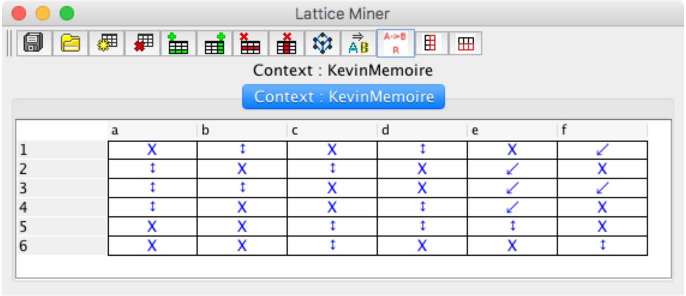


#	Antecedent	=>	Consequence	Support	Confidence
1.	{b, c}	=>	{f}	16.66%	100.0%
2.	{e}	=>	{a}	33.33%	100.0%
3.	{b, e}	=>	{d}	16.66%	100.0%
4.	{a, d}	=>	{e}	16.66%	100.0%
5.	{d, e}	=>	{b}	16.66%	100.0%
6.	{f}	=>	{b}	50.0%	100.0%
7.	{a, c}	=>	{e}	16.66%	100.0%

Les outils de l'analyse formelle de concepts

Lattice Miner

Figure 17: Relations flèches dans *Lattice Miner*



	a	b	c	d	e	f
1	X	↕	X	↕	X	↙
2	↕	X	↕	X	↙	X
3	↕	↕	X	X	↙	↙
4	↕	X	X	↕	↙	X
5	X	X	↕	↕	↕	X
6	X	X	↕	X	X	↕

Production d'implications avec négation

Production d'implications avec négation

Définition 15 : Catégories d'implications

$$r : Y \to Z \; [sup]$$

r est une implication du contexte $\mathbb{K}|\tilde{\mathbb{K}}$ tel que :

- Y et Z sont des sous-ensembles de $M \cup \tilde{M}$
- $Y \cap Z = \emptyset$

Les implications appartiennent à une des catégories suivantes :

- implications *purement négatives* avec $M \cap (Y \cup Z) = \emptyset$,
- implications *purement positives* avec $\tilde{M} \cap (Y \cup Z) = \emptyset$, et
- implications *mixtes* tel que $M \cap (Y \cup Z) \neq \emptyset$ et $\tilde{M} \cap (Y \cup Z) \neq \emptyset$.

Production d'implications avec négation

Définition 16 : Implications basées sur les clés (*key-based implication*)

$$r : Y \rightarrow M \cup \tilde{M} \backslash Y \ [0]$$

- Y une clé dans le contexte $\mathbb{K}|\tilde{\mathbb{K}}$, ce qui signifie que Y implique tous les attributs dans $M \cup \tilde{M}$
- Le support d'une telle implication est évidemment nul car il n'existe aucun itemset Y impliquant tous les attributs dans $M \cup \tilde{M}$.

Production d'implications avec négation

Solution

Théorème

Soit Ax avec x un attribut de $M\tilde{M}$ et $A \subseteq M\tilde{M}$. Alors, $Ax \rightarrow M\tilde{M} \backslash Ax$ [0] $\Leftrightarrow A \rightarrow \tilde{x}$ [sup] avec $sup = |A'|/|G|$ où A' est l'ensemble d'objets possédant les attributs contenus dans A et G est l'ensemble des objets du contexte $\mathbb{K}$.

Ces implications avec clé de la forme $Ax \rightarrow M\tilde{M} \backslash Ax$ ont un support nécessairement nul puisque même si un objet possède tous les attributs dans M, il n'en possède aucun dans $\tilde{M}$.

Production d'implications avec négation

Algorithme 1 : Génération des clés de l'infimum du treillis à partir de $\mathbb{K}$

Procédure
Entrée : Contexte initial $\mathbb{K} := (G, M, I)$
Sortie : $\mathcal{K}$: Les générateurs de l'infimum du treillis du contexte $\mathbb{K}|\tilde{\mathbb{K}}$
$INT \leftarrow \emptyset$ {Ensemble des intentions des co-atomes}
$O \leftarrow \emptyset$ {Objets traités}
while $G \neq O$ **do**
    for all $o \in G$ mais $o \notin O$ **do**
        $X \leftarrow o''$ {Calcul du fermé de l'objet o}
        $Y \leftarrow Intent(o) \cup (M \setminus Intent(o))^{\sim}$
        $O := O \cup X$
        $INT := INT \cup \{Y\}$
    end for
end while
$\mathcal{K} \leftarrow \mathrm{JEN}(M \cup \tilde{M}, INT)$
return $\mathcal{K}$

Production d'implications avec négation

	a	b	c	d	e
1	×				
2	×		×		
3	×	×			
4		×			
5	×		×		×
6	×	×	×		×
7		×	×	×	
8		×	×		×

Figure 18: Exemple de contexte $\mathbb{K}$

- L'exécution de l'algorithme 1 nous donne les clés suivantes : $\{a, a'\}$, $\{a', b'\}$, $\{b, b'\}$, $\{a, d\}$, $\{b', d\}$, $\{d, d'\}$, $\{c, c'\}$, $\{c', d\}$, $\{c', e\}$, $\{d, e\}$, $\{e, e'\}$, $\{a, b, c, e'\}$, $\{b, c, d', e'\}$, $\{a', c, d', e'\}$.
- Après avoir écarté les clés triviales représentant un attribut et sa négation (ex. $\{a, a'\}$), on retient les éléments suivants : $\{a', b'\}$, $\{a, d\}$, $\{b', d\}$, $\{c', d\}$, $\{c', e\}$, $\{d, e\}$, $\{a, b, c, e'\}$, $\{b, c, d', e'\}$, $\{a', c, d', e'\}$.

Production d'implications avec négation

Création des implications avec négation à partir des clés candidates

Candidats	$\{a', b'\}$	$\{a, d\}$	$\{b', d\}$
Règles	$\{b'\} \rightarrow \{a\}$ $\{a'\} \rightarrow \{b\}$	$\{d\} \rightarrow \{a'\}$ $\{a\} \rightarrow \{d'\}$	$\{d\} \rightarrow \{b\}$ $\{b'\} \rightarrow \{d'\}$
Candidats	$\{c', e\}$	$\{d, e\}$	$\{a, b, c, e'\}$
Règles	$\{e\} \rightarrow \{c\}$ $\{c'\} \rightarrow \{e'\}$	$\{e\} \rightarrow \{d'\}$ $\{d\} \rightarrow \{e'\}$	$\{b, c, e'\} \rightarrow \{a'\}$ $\{a, c, e'\} \rightarrow \{b'\}$ $\{a, b, e'\} \rightarrow \{c'\}$ $\{a, b, c\} \rightarrow \{e\}$
Candidats	$\{b, c, d', e'\}$	$\{a', c, d', e'\}$	$\{c', d\}$
Règles	$\{c, d', e'\} \rightarrow \{b'\}$ $\{b, d', e'\} \rightarrow \{c'\}$ $\{b, c, e'\} \rightarrow \{d\}$ $\{b, c, d'\} \rightarrow \{e\}$	$\{c, d', e'\} \rightarrow \{a\}$ $\{a', d', e'\} \rightarrow \{c'\}$ $\{a', c, e'\} \rightarrow \{d\}$ $\{a', c, d'\} \rightarrow \{e\}$	$\{d\} \rightarrow \{c\}$ $\{c'\} \rightarrow \{d'\}$

Production d'implications avec négation

Calcul du support des implications

Le support de chaque implication $Y \rightarrow Z$ se calcule avec la formule suivante :

$$Sup(T) = \sum_{PT(T) \subseteq U \subseteq T} (-1)^{n(U)} \times Sup(P(U)) \quad (1)$$

- $T = Y \cup Z$ représente l'ensemble de tous les attributs positifs et négatifs de l'implication
- $Sup(T)$ est le support de T,
- $PT(T)$ est l'ensemble des éléments positifs dans T,
- $n(T)$ est le nombre d'éléments négatifs dans T, et
- $P(U)$ représente l'ensemble des attributs dans U exprimés en éléments positifs.

Production d'implications avec négation

Intégration de la solution dans *Lattice Miner*

Figure 19: Affichage des implications avec négation

Association Rule Viewer

Context : ExempleNeg

Min. support : 0.0%
Min. confidence : 100.0%

Rule count : 24

#	Antecedent	=>	Consequence	Support	Confidence
1.	{b'}	=>	{a}	37.5%	100.0%
2.	{a'}	=>	{b}	37.5%	100.0%
3.	{d}	=>	{a'}	12.5%	100.0%
4.	{a}	=>	{d'}	62.5%	100.0%
5.	{d}	=>	{b}	12.5%	100.0%
6.	{b'}	=>	{d'}	37.5%	100.0%
7.	{d}	=>	{c}	12.5%	100.0%
8.	{c'}	=>	{d'}	37.5%	100.0%
9.	{e}	=>	{c}	37.5%	100.0%
10.	{c'}	=>	{e'}	37.5%	100.0%
11.	{e}	=>	{d'}	37.5%	100.0%
12.	{d}	=>	{e'}	12.5%	100.0%
13.	{b, c, e'}	=>	{a'}	12.5%	100.0%
14.	{a, c, e'}	=>	{b'}	12.5%	100.0%
15.	{a, b, e'}	=>	{c'}	12.5%	100.0%
16.	{a, b, c}	=>	{e}	12.5%	100.0%
17.	{c, d', e'}	=>	{b'}	12.5%	100.0%
18.	{b, d', e'}	=>	{c'}	25.0%	100.0%
19.	{b, c, e'}	=>	{d}	12.5%	100.0%
20.	{b, c, d'}	=>	{e}	25.0%	100.0%
21.	{c, d', e'}	=>	{a}	12.5%	100.0%
22.	{a', d', e'}	=>	{c'}	12.5%	100.0%

Enrichissement du module de génération des règles triadiques

Conclusion

Revue des contributions :

- Vérification des procédures de la base d'implications de Guigues-Duquenne ;
- Vérification des procédures de calcul des relations de flèches ;
- Production d'implications avec négation ;
- Enrichissement du module de génération des règles triadiques ;
- Mise à jour de Lattice Miner avec la dernière version de Java 8, support de Maven, et documentation technique ;
- Article pour la conférence ICFCA 2017 (Missaoui et Emamirad 2017) ;

Travaux futurs :

- Associer au module de calcul des implications (y compris celles avec négation et les implications triadiques) une autre procédure qui permet de calculer le fermé d'un groupe d'attributs Y avec ou sans un ensemble C de conditions ;
- Version Web *Lattice Miner* ;

References I

Jean-Louis Guigues et Vincent Duquenne. « Familles minimales d'implications informatives résultant d'un tableau de données binaires ». In : *Mathématiques et Sciences Humaines* 95 (1986), p. 5–18.

Bernhard Ganter et Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York, Inc., 1999.

Nicolas Pasquier et al. « Efficient Mining of Association Rules Using Closed Itemset Lattices ». In : *Information Systems* 24.1 (1999), p. 25–46.

Peter Becker, Joachim Hereth et Gerd Stumme. « ToscanaJ: An Open Source Tool for Qualitative Data Analysis ». In : *Advances in Formal Concept Analysis for Knowledge Discovery in Databases (FCAKDD'2002)*. Juil. 2002, p. 1–2.

References II

J. Pfaltz et C. Taylor. « Scientific Discovery through IterativeT ransformations of Concept Lattices ». In : *Proceedings of the 1st Internationla Workshop on Discrete Mathematics and Data Mining*. Avr. 2002, p. 65–74.

Bernhard Ganter et Sergei A. Obiedkov. « Implications in Triadic Formal Contexts ». In : *ICCS*. 2004, p. 186–195.

Laszlo Szathmary. « Symbolic Data Mining Methods with the Coron Platform ». Thèse de doctorat. Univ. Henri Poincaré - Nancy 1, France, nov. 2006.

Geneviève Roberge. « Visualisation des résultats d'une fouille de données dans les treillis de concepts ». Mém.de mast. Université du Québec en Outaouais, 2007.

References III

Christophe Demko et Karell Bertet. « Generation Algorithm of a Concept Lattice with Limited Object Access ». In : *Proceedings of The Eighth International Conference on Concept Lattices and Their Applications*. 2011, p. 239–250.

Rokia Missaoui, Lhouari Nourine et Yoan Renaud. « Computing Implications with Negation from a Formal Context ». In : *Fundam. Inf.* 115.4 (déc. 2012), p. 357–375.

Karell Bertet et al. *java-lattices: a Java library for lattices computation*. `http://thegalactic.org`. 2014.

Jean-François Viaud et al. « Décomposition sous-directe d'un treillis en facteurs irréductibles ». In : *Journées francophones d'Ingénierie des Connaissances IC 2015*. collection AFIA. Rennes, France, juin 2015.

References IV

Rokia Missaoui et Kévin Emamirad. « Lattice Miner 2.0: A Formal Concept Analysis Tool ». In : *Supplementary Proc. of ICFCA, Rennes, France, 2017*. 2017, p. 91–94.

Questions ?