Qualité d'un ensemble de règles : élimination des règles redondantes

Rémi Lehn*, Fabrice Guillet*, Henri Briand*

*Institut de Recherche en Informatique de Nantes
Université de Nantes
bâtiment IRESTE
La Chantrerie
44300 Nantes
{lehn,guillet,briand}@irin.univ-nantes.fr

Résumé. La qualité d'un ensemble de règles d'association est souvent considérée, par un utilisateur, selon la compréhension qu'il obtient du domaine étudié, en interprétant les règles qui lui sont présentées. Pour rendre l'ensemble de règles plus lisible, et donc améliorer ce critère de qualité global, nous appliquons aux règles d'association une méthode de réduction initialement proposée pour l'élimination des dépendances fonctionnelles redondantes. Malgré les différences entre les propriétés de ces deux types de relations, cette méthode permet d'obtenir des représentations de règles très concises et facilement interprétables par l'utilisateur.

1 Introduction

La découverte de règles d'association [Agrawal *et al.*, 1996] est motivée par l'utilisation de bases de données opérationnelles –c'est-à-dire dont la vocation principale est autre que de servir à des tâches d'ECD– pour découvrir une connaissance a priori inconnue et exploitable par un utilisateur dans un processus d'analyse ou de prise de décision [Lehn *et al.*, 1999]. Plusieurs algorithmes de découverte de règles d'association performants ont été publiés [Hipp *et al.*, 2000]. Une des hypothèses fondamentale de la découverte de règles d'association est que l'utilisateur de la connaissance produite ne spécifie pas de but à la découverte de connaissances. De par la nature intrinsèquement combinatoire de la méthode de découverte de règles et l'absence de buts a priori, l'utilisation classique de ces algorithmes (enchaînement des phases de sélection de données, mise en forme de ces données, induction de règles d'association, présentation des règles découvertes) fournit généralement un grand nombre de règles, sans aucun ordre, qui va donc à l'encontre du principe d'intelligibilité de la connaissance découverte par l'utilisateur inclus dans le processus d'ECD, et donc, influe directement sur la qualité de la connaissance perçue et effectivement exploitable par l'utilisateur dans son processus de prise de décision. Des expérimentations passées sur l'utilisation brute d'algorithmes de découverte de règles d'association comme l'algorithme *A Priori*, nous ont conduit à mettre en évidence des milliers de règles, à partir de bases de données dont le volume était comparable à celui des règles produites. Quelle est alors la qualité de la vision que l'utilisateur obtient du domaine étudié, s'il doit explorer des milliers de règles ? Quelle est la qualité de la procédure d'induction elle-même, lorsque l'effort à mettre en œuvre

par l'utilisateur pour interprétater les règles devient comparable à l'effort nécessaire pour obtenir la même compréhension du domaine, à partir de l'observation des données elles-mêmes ?

Une réponse classique à ce problème est de fixer des seuils élevés sur les indices de qualités, mesurés pour chaque règle, afin d'éliminer les règles les moins pertinentes selon les hypothèses traduites par ces indices. Il existe cependant des cas où cette réponse est inadéquate : lorsque l'utilisateur ne sait pas fixer les seuils correspondant à la connaissance qu'il recherche, ou bien lorsqu'il recherche des règles ayant des propriétés en dehors de celles modélisées par les indices à sa disposition ou encore lorsqu'il existe un grand nombre de dépendances cachées dans les données. Cette inadéquation se traduira souvent, dans les résultats d'approches classiques en découverte de règles d'association, par la production d'un très grand nombre de règles, un volume trop important pour être interprété tel quel par l'utilisateur. De plus, des critères globaux d'interprétation peuvent être ajoutés, en plus de ceux mesurés, sur des règles individuelles :

- des critères d'opérationnalité, pour des tâches de décision précises [Brachman et Anand, 1996], pour lesquelles il n'existe pas forcément d'indices de qualité appropriés pour chaque règle.
- des critères d'intelligibilité, dont l'évaluation précise repose sur une qualification cognitive de la perception de l'utilisateur de la connaissace représentée, et de l'ergonomie de cette représentation et de sa visualisation, vis à vis des tâches de décision ciblées. Néanmoins, sous l'hypothèse d'une lecture linéaire (par opposition à l'utilisation d'une interface de visualisation interactive adaptée) de l'ensemble des règles d'association par l'utilisateur, la limitation du volume de règles, associée à une convention de lecture, constitue un facteur important dans l'amélioration de ces critères de qualité, reposant sur l'intelligibilité.
- Des critères liés à l'exploitation des règles d'association par des procédures automatisés (moteurs d'inférences, par exemple). Dans ce cas, le respect d'un comportement inférenciel particulier des règles (logique d'ordre 0 par exemple) constitue une nouvelle base d'évaluation de la qualité de la connaissance.

Nous développerons donc ici une approche alternative et complémentaire à l'évaluation individuelle de chaque règle. Elle consiste à limiter la quantité de règles d'association, afin d'en améliorer l'intelligibilité, en ne représentant pas à l'utilisateur les règles que celui-ci pourrait obtenir lui-même, par raisonnement, à partir des règles représentées. Les règles ainsi éliminées sont alors considérées comme redondantes par rapport aux autres règles représentées. Cette approche considère un critère global de qualité de la connaissance produite (son intelligibilité), complétant la mesure de la qualité de chaque règle, considérée individuellement.

La méthode d'élimination de règles redondantes dépend fortement de la définition d'un modèle de représentation, incluant notamment une définition d'un modèle des règles que l'utilisateur peut déduire lui-même à partir des règles représentées (convention de lecture). Plusieurs modèles ont été développés ; certains sont couplés à des algorithmes de découverte de règles [Guigues et Duquennes, 1986] [Ganascia, 1987] [Fleury, 1994] ; d'autres sont liés à des modèles de représentation particuliers, notamment les treillis de Galois [Dumitriu *et al.*, 2000][Pasquier *et al.*, 1999][Pasquier, 2000] [Boulicaut *et al.*, 2000].

Le modèle de représentation que nous développerons ici est basé sur des propriétés logiques (du calcul propositionnel), sous l'hypothèse d'un comportement implicatif des règles d'association. Nous commencerons par présenter une méthode que nous avons utilisée, inspirée par la méthode d'élimination de dépendances fonctionnelles redondantes par calcul de la couverture minimale. Nous étendrons ensuite cette méthode aux implications logiques ; nous la comparerons aux autres représentations de règles. Nous terminerons en présentant son application et ses limites pour représenter des ensembles de règles d'association. Dans cet exemple, nous détaillerons notamment le comportement de la méthode de réduction de l'ensemble de règles comme un traitement suivant un filtrage individuel des règles au moyen d'indices de qualité.

2 Dépendances fonctionnelles redondantes

D'importants travaux ont déjà été réalisés pour éliminer les dépendances fonctionnelles redondantes dans les bases de données relationnelles (par exemple, dans la communauté des bases de données, ceux d'Ullman [Ullman, 1982] [Ullman, 1989a] [Atkins, 1988] [Delobel et Adiba, 1982] [Briand *et al.*, 1986] ; il existe aujourd'hui des algorithmes permettant de découvrir automatiquement une représentation des dépendances fonctionnelles sur un ensemble de relations stockées dans une base de données (par exemple l'algorithme *Dep-Miner* [Lopes *et al.*, 1999]).

2.1 Définitions

2.1.1 Dépendances fonctionnelles

Il existe une dépendance fonctionnelle (D.F.), notée $A \rightarrow B$ entre un sous ensemble A des attributs d'une relation R et un autre sous ensemble B des attributs de la relation R si et seulement si la relation R associe un et un seul ensemble de valeurs aux attributs de l'ensemble B pour chaque ensemble de valeurs possibles pour les attributs de l'ensemble A [Mannila et Räihä, 1992][Ullman, 1989a].

2.1.2 Axiomes d'Armstrong

Les trois axiomes de base du calcul sur les systèmes de dépendances fonctionnelles sont les axiomes d'Armstrong [Ullman, 1989b] :

Réflexivité : $\vdash\!\!\!> A \cup B \rightarrow A.$ (1)

Augmentation : $A \rightarrow B \vdash\!\!\!> A \cup C \rightarrow B \cup C.$ (2)

Transitivité : $A \rightarrow B, B \rightarrow C \vdash\!\!\!> A \rightarrow C.$ (3)

2.1.3 Formes redondantes

[Briand *et al.*, 1986] définissent des formes redondantes usuelles (table 1), qui sont des théorèmes démontrables à partir des axiomes d'Armstrong.

Nom	D.F.		D.F. redondantes	
Augmentation	$A \rightarrow B$	$\mapsto$	$A \cup C \rightarrow B.$	(4)
Transitivité	$A \rightarrow B$ $B \rightarrow C$	$\mapsto$	$A \rightarrow C.$	(5)
1ère forme de pseudo-transitivité	$A \rightarrow B$ $B \cup C \rightarrow D$	$\mapsto$	$A \cup C \rightarrow D.$	(6)
Réflexivité		$\mapsto$	$A \rightarrow A.$	(7)
Union	$A \rightarrow B$ $A \rightarrow C$	$\mapsto$	$A \rightarrow B \cup C.$	(8)
Décomposition	$A \rightarrow B \cup C$	$\mapsto$	$A \rightarrow B$ $A \rightarrow C.$	(9)
2ème forme de pseudo-transitivité	$A \rightarrow B$ $A \cup B \rightarrow C$	$\mapsto$	$A \rightarrow C.$	(10)

TAB. 1 – Formes redondantes usuelles dans les systèmes de dépendances fonctionnelles.

2.1.4 Fermeture d'un ensemble de D.F.

La fermeture d'un ensemble de D.F. noté $F = (A, \cup, \rightarrow)$ définies sur un ensemble d'attributs A est l'ensemble F^+ qu'il est possible d'écrire à partir de F en utilisant les axiomes d'Armstrong (définition de Ullman, 1989 [Ullman, 1989b], reprise par Fleury, 1996 [Fleury, 1996]).

2.1.5 Equivalence de deux ensembles de D.F.

Deux ensembles de D.F. F_1 et F_2 sont équivalents si

$$F_1^+ = F_2^+ \tag{11}$$

2.1.6 Fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F.

La fermeture d'un ensemble d'attributs $X \subset A$ par rapport à un ensemble de D.F. $F = (A, \cup, \rightarrow)$ est l'ensemble d'attributs

$$X_F^+ = \cup_i \{Y_i \mid (X \rightarrow Y_i) \in F^+\} \tag{12}$$

(c'est également une définition de Ullman, 1989 [Ullman, 1989b], reprise par Fleury, 1996 [Fleury, 1996]).

2.2 Décomposition des dépendances fonctionnelles

Les réécritures d'*union* et de *décomposition*, (8) et (4) sur la table 1, page 4 permettent de définir l'équivalence

$$\{A \rightarrow B\} \equiv \{A \rightarrow \{b\} \mid b \in B\}. \tag{13}$$

Entrée : F_0, un ensemble de D.F.
Sortie : F_1, un ensemble de D.F. équivalent à F_0, mais dont les parties droites de D.F. sont des singletons.

```
1  F_1 = ∅
2  pour chaque (X → Y) ∈ F_0 faire
3  |  pour chaque y ∈ Y faire
4  |  |  F_1 = F_1 ∪ {X → {y}}
```

Algorithme 1: Décomposition des D.F.

La décomposition des dépendances fonctionnelles consiste à réécrire l'ensemble des D.F. sous la forme de la partie droite de l'équivalence (13). L'intérêt de décomposer les D.F. est double : éliminer les redondances dues à l'union (8) et à la décomposition (9) d'une part, et simplifier les traitements sur l'ensemble de D.F. d'autre part [Briand *et al.*, 1986][Ullman, 1989b]. L'algorithme 1 permet de réaliser cette décomposition.

De même, l'écriture sous la forme de la partie gauche de l'équivalence (13) permet une représentation plus compacte des D.F. après l'élimination des D.F. redondantes. (algorithme 2).

Entrée : F_0, un ensemble de D.F.
Sortie : F_1, un ensemble de D.F. équivalent à F_0, mais dont chaque partie gauche de D.F. est unique.

```
5  F_1 = ∅
6  Soit F'_0 = F_0.
7  Soit X' = ∅.
8  Soit Y' = ∅.
9  Trier les D.F. de F'_0 par ordre croissant des parties gauche de D.F.
10 pour chaque (X → Y) ∈ F'_0 faire
11 |  si X == X' alors
12 |  |  Y' = Y' ∪ Y.
   |  sinon
13 |  |  si X' ≠ ∅ alors
14 |  |  |  F_1 = F_1 ∪ {X' → Y'}.
15 |  |  X' = X.
16 |  |  Y' = Y.
17 F_1 = F_1 ∪ {X' → Y'}.
```

Algorithme 2: Union des D.F. ayant une partie gauche commune.

3 Algorithme de calcul de la couverture minimale

3.1 Couverture minimale

La méthode de la recherche de la couverture minimale, également référencée sous le nom algorithme d'Ullman car publiée par cet auteur [Ullman, 1989a], permet de rechercher un ensemble minimal de D.F., noté $\hat{F}$, d'un ensemble de D.F. F, tel que

$$\hat{F}^+ = F^+ \text{ et } \hat{F} \text{ minimal.} \tag{14}$$

$\hat{F}$ est minimal s'il ne contient ni *D.F. redondante*, ni *attribut superflu*.

3.1.1 D.F. redondante

Une D.F. est dite redondante si elle peut être calculée par les axiomes d'Armstrong sur le système de D.F. privée de cette D.F. : $X \rightarrow Y$ de F est redondante si

$$F \subset (F \setminus \{X \rightarrow Y\})^+. \tag{15}$$

Pour vérifier cette condition, il suffit que

$$(X \rightarrow Y) \in (F \setminus \{X \rightarrow Y\})^+. \tag{16}$$

En appliquant la définition (12) de la fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F., on peut la définir comme étant redondante si

$$Y \subset X^+_{(F \setminus \{X \rightarrow Y\})}. \tag{17}$$

Ullman [Ullman, 1989b] montre que si $Y \subset X^+_F$, alors $(X \rightarrow Y) \in F^+$.

3.1.2 Attribut superflu

Un attribut x de la partie gauche d'une D.F. $X \rightarrow Y$ est superflu si la D.F. $(X \setminus x) \rightarrow Y$ peut être calculée par les axiomes d'Armstrong sur le système de D.F. ; c'est-à-dire si

$$F \subset ((F \setminus \{X \rightarrow Y\}) \cup \{(X \setminus x) \rightarrow Y\})^+. \tag{18}$$

Pour vérifier cette condition, il suffit que

$$((X \setminus x) \rightarrow Y) \in F^+ \tag{19}$$

ou

$$Y \subset (X \setminus x)^+_F. \tag{20}$$

Entrée : Un ensemble de dépendances fonctionnelles F.

Sortie : Une couverture minimale $\hat{F}$ de F.

18 $\hat{F} = F$.
19 *minimal* = faux.
20 **tant que** *non minimal* **faire**
21 *minimal* = vrai.
22 **pour chaque** $(X \rightarrow Y)$ = *choix*($\hat{F}$) **faire**
23 **pour chaque** *action* = *choix*({*attribut, df*}) **faire**
24 **si** *action* == *df* **alors**
25 **si** $Y \subset X^{+}_{(\hat{F} \setminus \{X \rightarrow Y\})}$ **alors**
26 $\hat{F} = (\hat{F} \setminus \{X \rightarrow Y\})$.
27 *minimal* = faux.
 sinon
28 **pour chaque** $x \in X$ **faire**
29 **si** $Y \subset (X \setminus x)^{+}_{\hat{F}}$ **alors**
30 $\hat{F} = (\hat{F} \setminus \{X \rightarrow Y\}) \cup \{(X \setminus x) \rightarrow Y\}$.
31 *minimal* = faux.

Algorithme 3: Couverture minimale : algorithme non déterministe.

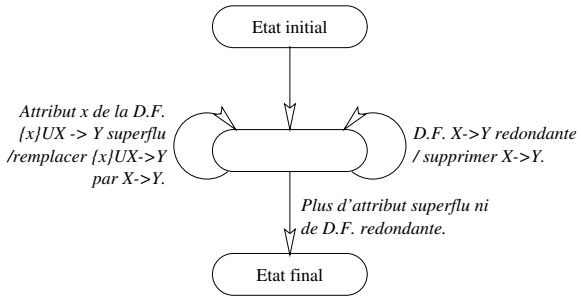


FIG. 1 – Automate de calcul de la couverture minimale.

$F_0 = \{A, B \to C, C \to B, A \to B\}$.
$C \notin (\{A, B\})^+_{(F_0 \setminus \{A, B \to C\})}$.
$B \notin C^+_{(F_0 \setminus \{C \to B\})}$.
$B \notin A^+_{(F_0 \setminus \{A \to B\})}$.
$\Rightarrow$ pas de D.F. redondante.

$C \notin (\{A, B\} \setminus A)^+_{F_0}$.
$C \in (\{A, B\} \setminus B)^+_{F_0}$.
$\Rightarrow$ L'attribut B de la partie gauche de la D.F. $A, B \to C$ est superflu.

ATTENTION : $(F_0 \setminus \{A, B \to C\}) \cup \{A \to C\}$ n'est pas une couverture minimale de F_0 : la D.F. $A \to B$ est redondante !

TAB. 2 – Contre-exemple d'Atkins.

3.2 Algorithmes proposés

L'application directe des définitions de D.F. redondantes (17) et d'attribut superflus permet d'écrire l'automate de la figure 1, qui correspond à la définition "*choisir une D.F., si elle est redondante, la supprimer ; ou si elle contient un attribut superflu, le supprimer, et recommencer jusqu'à ce qu'il n'y ait plus ni D.F. redondante, ni attribut superflu*". Cet automate, non-déterministe à l'état intermédiaire sur le choix de la D.F. à évaluer et sur le choix de l'action à entreprendre (chercher à supprimer une D.F. redondante ou un attribut superflu), peut être retranscrit en l'algorithme 3. L'état intermédiaire correspond à une itération de la boucle principale de l'algorithme (lignes 21 à 31) pendant laquelle la suppression d'une D.F. redondante (ligne 25) ou d'un attribut superflu (ligne 29) peut intervenir. On peut remarquer que cet algorithme ne calcule jamais la fermeture F^+.

De par leur définition, la suppression d'une D.F. redondante ou d'un attribut superflu conserve la fermeture. La première instruction *choix* de l'algorithme 3 (ligne 22) peut donc être remplacée par un simple parcours itératif de l'ensemble de D.F.

22 **pour chaque** $(X \to Y) \in \hat{F}$ ... **faire**

La suppression de la deuxième instruction *choix* de l'algorithme 3 (ligne 22) est plus problématique ; en effet, Atkins [Atkins, 1988] montre que l'élimination de toutes les D.F. redondantes –on obtient alors un ensemble de D.F. ne comportant plus aucune D.F. redondante– puis l'élimination de tous les attributs superflus peut conduire à une couverture non minimale. Il donne le contre-exemple de la table 2 montrant qu'à l'issue de l'élimination des attributs superflus sur un ensemble de D.F. ne comportant pas de D.F. redondante, on peut obtenir à nouveau des D.F. redondantes.

Par contre, Atkins montre que l'élimination des D.F. redondantes sur un ensemble de D.F. ne comportant pas d'attribut superflu conduit bien à une couverture minimale. Grâce à cette remarque, l'algorithme 3, non déterministe, peut être réécrit sous une

forme déterministe (algorithme 4), l'étape d'élimination des attributs superflus (lignes 32 à 39) précédant l'élimination des D.F. redondantes (lignes 40 à 43).

Entrée : Un ensemble de dépendances fonctionnelles F.

Sortie : Une couverture minimale $\hat{F}$ de F.

```
32 Soit F_k = ∅.
33 pour chaque (X → Y) ∈ F faire
34     Soit X_k = ∅.
35     pour chaque x ∈ X faire
36         si Y ⊄ (X \ x)+_F alors
37             X_k = X_k ∪ {x}.
38     si X_k ≠ ∅ alors
39         F_k = F_k ∪ {X_k → Y}.
40 F̂ = F_k.
41 pour chaque (X → Y) ∈ F_k faire
42     si Y ⊂ X+_(F̂\{X→Y}) alors
43         F̂ = (F̂ \ {X → Y}).
```

Algorithme 4: Couverture minimale : algorithme déterministe.

L'algorithme 4 de recherche de la couverture minimale ne nécessite que la détermination de l'inclusion d'un ensemble d'attributs dans la fermeture d'un autre ensemble d'attributs par rapport à un ensemble de D.F. (lignes 36 et 42 de l'algorithme 4). En décomposant les D.F. pour obtenir un ensemble de D.F. dont les parties droites sont des singletons (équivalence (13) à la page 4 et algorithme 1 à la page 5), il est même suffisant de ne déterminer que l'appartenance d'un seul attribut à la fermeture.

Le calcul de la fermeture F^+ (problème exponentiel par nature[1]) peut donc être évité s'il est possible de déterminer facilement l'appartenance d'un attribut à la fermeture d'un autre ensemble d'attributs par rapport à un ensemble de D.F.

3.2.1 Détermination de l'appartenance d'un attribut à la fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F.

L'axiome d'Armstrong de réflexivité (1) permet d'écrire

$$\begin{array}{rcl} & \vdash\!\!\!\rightarrow & (A \cup A) \rightarrow A \\ (A \cup A) \rightarrow A & \vdash\!\!\!\rightarrow & A \rightarrow A; \end{array} \tag{21}$$

donc

$$y \in X_F^+ \ \underline{si}\ y \in X; \tag{22}$$

[1] le seul axiome de réflexivité produisant $n \times 2^{n-1}$ D.F. avec n attributs, le calcul de la fermeture ne peut pas être dans la classe P.

car x sera bien en partie droite d'une D.F. appartenant à F^+ et dont la partie gauche est incluse dans $\{x\}$.

A partir d'une D.F. $(A \rightarrow B) \in F$, l'axiome d'Armstrong d'augmentation (2) permet d'écrire

$$\begin{array}{lll} A \rightarrow B & \vdash\!\!> & (A \cup A) \rightarrow (A \cup B) \\ et\ (A \cup A) \rightarrow (A \cup B) & \vdash\!\!> & A \rightarrow (A \cup B); \end{array} \tag{23}$$

or, si on ajoute la D.F. $((A \cup B) \rightarrow C) \in F$, on peut écrire, grâce à l'axiome d'Armstrong de transitivité (3)

$$\left.\begin{array}{l} A \rightarrow (A \cup B)\ \ (23) \\ (A \cup B) \rightarrow C \end{array}\right\} \vdash\!\!> A \rightarrow C. \tag{24}$$

Il en est de même avec toute réécriture utilisant les axiomes d'Armstrong dont la partie gauche est incluse dans l'ensemble $A \cup B$. La seule démonstration $B \subset A_F^+$ suffit à déterminer que si $(A \cup B \rightarrow C) \in F$, alors $C \subset (A \cup B)_F^+$; de plus aucune réécriture supplémentaire que permet $A \rightarrow B$ ne conduira à des parties droites de D.F. ne contenant que des sous-ensembles de $A \cup B$; donc,

$$y \in X_F^+ \ \underline{si}\ \exists A \rightarrow B \in F \mid A \subset X\ \underline{et}\ y \in (X \cup A)_{(F \setminus \{A \rightarrow B\})}. \tag{25}$$

La détermination de l'appartenance d'un attribut à la fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F. peut donc s'écrire

$$y \in X_F^+ \ \underline{si}\ \left\{\begin{array}{ll} & y \in X \\ \underline{ou} & \exists A \rightarrow B \in F \mid A \subset X\ \underline{et}\ y \in (X \cup A)^+_{(F \setminus \{A \rightarrow B\})}. \end{array}\right. \tag{26}$$

Cette définition récursive (récursivité terminale) peut être traduite de manière itérative (algorithme 5).

3.3 Exemple

Le tableau 2 présente l'élimination des attributs superflus, en déroulant l'algorithme pas à pas sur un exemple simple. Les tableaux 3 et 4 présentent de la même manière le déroulement de l'algorithme d'élimination des règles redondantes.

3.4 Complexité des algorithmes

Hypothèse : les opérateurs ensemblistes (union, sous-ensemble, appartenance à un ensemble) ont une complexité proportionnelle à la cardinalité minimum des ensembles sur lesquels ils opèrent.

Notations : N_F est le nombre de D.F., N_A est la cardinalité de l'ensemble des attributs de F, $N_{A_{A \rightarrow B}}$ est le nombre d'attributs dans la partie gauche de la D.F. $A \rightarrow B$, $N_{B_{A \rightarrow B}}$ est le nombre d'attributs de la partie droite de $A \rightarrow B$, $N_{A_F} = \sum_{A \rightarrow B \in F} N_{A_{A \rightarrow B}}$ et $N_{B_F} = \sum_{A \rightarrow B \in F} N_{B_{A \rightarrow B}}$.

La décomposition des D.F. (algorithme 1 à la page 5) a une complexité de $O(N_{B_F})$.

Entrée : – F : un ensemble de D.F.
– X : un ensemble d'attributs.
– y : un attribut.

Sortie : une valeur booléenne : vrai si $y \in X_F^+$, faux sinon.

```
44 Soit F_i = F.
45 Soit X_i = X.
46 Soit F_k = ∅.
47 fermé = faux.
48 tant que non fermé et y ∉ X_i faire
49 |  fermé = vrai.
50 |  F_k = ∅.
51 |  pour chaque A → B ∈ F_i faire
52 |  |  si A ⊂ X_i alors
53 |  |  |  X_i = X_i ∪ B.
54 |  |  |  fermé = faux.
   |  |  sinon
55 |  |  |  F_k = F_k ∪ {A → B}.
56 |  |  F_i = F_k.
57 si y ∈ X_i alors
58 |  y ∈ X_F^+ !
   sinon
59 |  y ∉ X_F^+ !
```

Algorithme 5: Détermination de l'appartenance d'un attribut à la fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F.

32	attributs superflus : $F = \{\ a \to b, a \wedge c \to b\ \}$
34	attributs superflus : $(X \to Y) = (a \to b)$
34	attributs superflus : $(X \to Y) = (a \wedge c \to b)$
36	attributs superflus : $(X \setminus x) = (\{a, c\} \setminus \{a\})$
	fermeture : $b \in X_F^+$?
	fermeture : $F = \{a \to b, a \wedge c \to b\}$, $X = c$
46	fermeture : $F_k = \emptyset$
51	fermeture : $(A \to B) = (a \to b)$
55	fermeture : $A \not\subset X_i$
	fermeture : $F_k = \{a \to b\}$
	fermeture : $X_i = \{c\}$
51	fermeture : $(A \to B) = (a \wedge c \to b)$
55	fermeture : $A \not\subset X_i$
	fermeture : $F_k = \{a \to b, a \wedge c \to b\}$
	fermeture : $X_i = \{c\}$
59	fermeture : $b \notin X_F^+$!
36	attributs superflus : $a \notin (\{c\})_F^+$
36	attributs superflus : $(X \setminus x) = (\{a, c\} \setminus \{c\})$
	fermeture : $b \in X_F^+$?
	fermeture : $F = \{a \to b, a \wedge c \to b\}$, $X = a$
46	fermeture : $F_k = \emptyset$
51	fermeture : $(A \to B) = (a \to b)$
53	fermeture : $A \subset X_i$
	fermeture : $F_k = \{\}$
	fermeture : $X_i = \{a, b\}$
51	fermeture : $(A \to B) = (a \wedge c \to b)$
55	fermeture : $A \not\subset X_i$
	fermeture : $F_k = \{a \wedge c \to b\}$
	fermeture : $X_i = \{a, b\}$
56	fermeture : $F_i = F_k$
	fermeture : $F_i = \{a \wedge c \to b\}$
58	fermeture : $b \in X_i$!
	fermeture : $b \in X_F^+$!
36	attributs superflus : $c \in (\{a\})_F^+$
36	attributs superflus : c est superflu.
39	attributs superflus : $(X_k \to Y) = (a \to b)$

FIG. 2 – Exemple d'application de l'algorithme d'élimination des redondances sur le sous-ensemble de règles $\{a \to b, a \wedge c \to b\}$ (élimination des attributs superflus). Les numéros de lignes de réfèrent aux différents algorithmes.

40 DF. redondantes : $\hat{F} = F_k = \{\ a \to b, a \to c, b \to c\ \}$
42 DF. redondantes : $(X \to Y) = (a \to b)$
42 DF. redondantes : $\hat{F} \setminus (X \to Y) = \{\ a \to c, b \to c\ \}$
fermeture : $b \in X_F^+$?
fermeture : $F = \{b \to c, a \to c\}$, $X = a$
46 fermeture : $F_k = \emptyset$
51 fermeture : $(A \to B) = (b \to c)$
55 fermeture : $A \not\subset X_i$
fermeture : $F_k = \{b \to c\}$
fermeture : $X_i = \{a\}$
51 fermeture : $(A \to B) = (a \to c)$
53 fermeture : $A \subset X_i$
fermeture : $F_k = \{b \to c\}$
fermeture : $X_i = \{a, c\}$
56 fermeture : $F_i = F_k$
fermeture : $F_i = \{b \to c\}$
46 fermeture : $F_k = \emptyset$
51 fermeture : $(A \to B) = (b \to c)$
55 fermeture : $A \not\subset X_i$
fermeture : $F_k = \{b \to c\}$
fermeture : $X_i = \{a, c\}$
59 fermeture : $b \notin X_F^+$!
42 DF. redondantes : $b \notin (\{a\})^+_{(\hat{F} \setminus (X \to Y))}$!
43 DF. redondantes : $\hat{F} = \{\ a \to b, a \to c, b \to c\ \}$
42 DF. redondantes : $(X \to Y) = (b \to c)$
42 DF. redondantes : $\hat{F} \setminus (X \to Y) = \{\ a \to b, a \to c\ \}$
fermeture : $c \in X_F^+$?
...
(de manière similaire, on montre que :)
42 DF. redondantes : $c \notin (\{b\})^+_{(\hat{F} \setminus (X \to Y))}$!
43 DF. redondantes : $\hat{F} = \{\ a \to b, a \to c, b \to c\ \}$

FIG. 3 – Exemple d'application de l'algorithme d'élimination des redondances sur le sous-ensemble de règles $\{a \to b, b \to c, a \to c\}$ (élimination des règles redondantes) : examen des règles $a \to b$ et $a \to c$.

42	DF. redondantes : $(X \rightarrow Y) = (a \rightarrow c)$
42	DF. redondantes : $\hat{F} \setminus (X \rightarrow Y) = \{\ a \rightarrow b, b \rightarrow c\ \}$
	fermeture : $c \in X_F^+$?
	fermeture : $F = \{a \rightarrow b, b \rightarrow c\}$, $X = a$
46	fermeture : $F_k = \emptyset$
51	fermeture : $(A \rightarrow B) = (a \rightarrow b)$
53	fermeture : $A \subset X_i$
	fermeture : $F_k = \{\}$
	fermeture : $X_i = \{a, b\}$
51	fermeture : $(A \rightarrow B) = (b \rightarrow c)$
53	fermeture : $A \subset X_i$
	fermeture : $F_k = \{\}$
	fermeture : $X_i = \{a, b, c\}$
56	fermeture : $F_i = F_k$
	fermeture : $F_i = \{\}$
58	fermeture : $c \in X_i$!
	fermeture : $c \in X_F^+$!
42	DF. redondantes : $c \in (\{a\})^+_{\hat{F} \setminus (X \rightarrow Y)}$!
	DF. redondantes : $a \rightarrow c$ est redondante !
43	DF. redondantes : $\hat{F} = \{\ a \rightarrow b, b \rightarrow c\ \}$

FIG. 4 – Exemple d'application de l'algorithme d'élimination des redondances sur le sous-ensemble de règles $\{a \rightarrow b, b \rightarrow c, a \rightarrow c\}$ (élimination des règles redondantes) : examen de la règle $b \rightarrow c$.

L'union des D.F. ayant une partie droite unique (algorithme 2 à la page 5) a une complexité de $O(N_{A_F} \times log(N_F))$; c'est en fait un tri + dédoublonnage, la comparaison de deux éléments ayant une complexité de $O(N_{A_F}/N_F)$.

La détermination de l'appartenance d'un attribut à la fermeture d'un ensemble d'attributs par rapport à un ensemble de D.F. est réalisée par un algorithme glouton (algorithme 5 à la page 11) ; à chaque pas de l'algorithme (lignes 48 à 56), une ou plusieurs D.F. sont consommées pour mettre à jour la fermeture partielle X_i (lignes 53 et 54) et, dans le cas où la condition d'arrêt ($y \in X_i$) ne peut pas être vérifiée et qu'au moins une D.F. a pu être consommée, l'algorithme reprend une nouvelle itération avec au moins une D.F. de moins.

Dans le pire des cas, une seule D.F. peut être consommée par itération. La détermination des D.F. de F_i pouvant être consommées à chaque itération (lignes 51 à 55) demandant N_{F_i} itérations, on aura donc, dans le pire des cas, $N_F + (N_F - 1) + (N_F - 2) + ... + 1$, soit $(N_F \times (N_F + 1))/2$ itérations. Chaque itération ayant une complexité de N_{A_F}/N_F, la complexité de l'algorithme 5 est donc de $O(N_{A_F} \times N_F)$ dans le pire des cas[2].

L'élimination des attributs superflus (lignes 32 à 39 de l'algorithme 4 à la page 9) est une itération sur les D.F. de F (lignes 33 à 39) ; pour chaque D.F., chaque attribut en partie gauche est testé. On aura donc N_{A_F} exécutions du test d'appartenance à la fermeture (ligne 36). Celle-ci pouvant être établie avec une complexité de $O(N_{A_F} \times N_F)$, la complexité de l'algorithme d'élimination des attributs superflus est donc de $O(N_{A_F}^2 \times N_F)$ dans le pire des cas.

L'élimination des D.F. redondantes (lignes 40 à 43 de l'algorithme 4 à la page 9) est une itération sur les D.F. de F_k. Si aucun atome superflu n'a été éliminé, le nombre d'itérations sera donc N_F[3]. Dans le pire des cas, aucune D.F. n'est redondante et la complexité de l'algorithme d'élimination des D.F. redondante est de $O(N_{A_F} \times N_F^2)$.

4 Comparaison avec d'autres modèles de représentation d'implications

Les axiomes d'Armstrong sont fermés et complets [Ullman, 1989b] et sont des théorèmes du calcul propositionnel. Ces propriétés permettent de montrer que d'une part toutes les expressions qu'il est possible de former en utilisant les axiomes d'Armstrong sont vraies pour la logique d'ordre 0 et d'autre part que l'application des axiomes d'Armstrong à saturation, c'est-à-dire jusqu'à arriver à un état du système formel où plus aucun axiome n'est applicable sans former une expression déjà existante dans le système, conduit à un l'ensemble des expressions valides qu'il est possible de construire à partir du système de dépendances fonctionnelles ; Laurent Fleury l'appelle "système redondant maximum" dans sa thèse [Fleury, 1996], et c'est également la fermeture (F^+) définie par Ullman [Ullman, 1989a].

[2] elle est de $O(1)$ dans le meilleur des cas, lorsque $y \in X$.

[3] dans le meilleur des cas on n'aura plus qu'une seule D.F. après élimination des attributs superflus.

Kaufmann démontre que les formules utilisées pour l'élimination des dépendances fonctionnelles dans les bases de données sont vraies également pour les implications logiques[Kaufmann, 1987].

4.1 Equivalence des systèmes formels

Ullman appuie ses démonstrations sur les propriétés des D.F. rendant valide le calcul propositionnel sur les D.F. [Ullman, 1989b]. Fleury, dans sa thèse [Fleury, 1996] et Kaufmann dans son livre [Kaufmann, 1987] montrent que l'algorithme de recherche de la couverture minimale est valide dans le cas des implications logiques. Il existe donc une analogie entre un système de D.F. $F = (A, \cup, \rightarrow)$ et un monde en calcul propositionnel $w = (A, \wedge, \rightarrow)$, avec A un ensemble de propositions, $\wedge$, la conjonction logique et $\rightarrow$, l'implication logique. Les axiomes d'Armstrong peuvent s'écrire alors sous la forme

réflexivité : $\vdash\!\!\!> a \wedge b \rightarrow a,$ (27)

augmentation : $a \rightarrow b \vdash\!\!\!> a \wedge c \rightarrow b \wedge c,$ (28)

transitivité : $a \rightarrow b, b \rightarrow c \vdash\!\!\!> a \rightarrow c,$ (29)

et sont des théorèmes du calcul propositionnel.

4.2 Comparaison avec les treillis conceptuels

La couverture minimale et les treillis conceptuels ont en commun l'utilisation des propriétés d'inclusion dans l'espace de représentation des extensions (implications logiques dans l'espace de représentation des intentions) pour limiter le volume de connaissances représentées.

4.2.1 Règles d'association et treillis de Galois

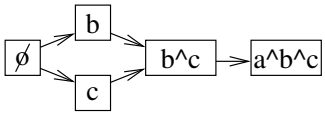
Les implications logiques utilisées par le calcul de la couverture minimale se traduisent par des pseudo-intentions (descriptions non fermées) n'apparaissant pas sur le treillis conceptuel. Globalement, plus le nombre d'implications logiques est important, plus le treillis conceptuel est petit et inversement. Dans le cadre de la représentation de règles d'association, l'utilisation du treillis conceptuel permet de rendre les règles logiques implicites [Pasquier, 2000] : elles n'apparaissent pas sur la représentation, par contre, elles peuvent être déduites à partir des connaissances représentées (exemple figure 5). Le treillis de Galois permet de représenter en plus les autres règles d'association (non logiques). Il est démontré [Pasquier, 2000][Dumitriu *et al.*, 2000] que la représentation des règles d'association sur un treillis de Galois permet de déduire l'ensemble des règles d'association :

1. le support d'une description fréquente non fermée (pseudo-intention), non représentée sur le treillis de Galois et égal au support de la description fréquente fermée minimale l'incluant (dans l'exemple de la figure 5, le support de a (non fermé) est égal au support de $a \wedge b \wedge c$ (fermé)) [Pasquier, 2000][Dumitriu *et al.*, 2000] ;
2. toute description sur la frontière de l'exploration est fermée [Pasquier, 2000] ;

❶ ***Relation :***

a	b	c
X	X	X
	X	X
		X
	X	

❹ ***Treillis de Galois :***



❸ ***Couverture minimale :*** a–>b^c

❷ ***33 règles logiques :***

	a	b	c	a^b	a^c	b^c	a^b^c
a	X			X	X		X
b	X	X		X	X	X	X
c	X		X	X	X	X	X
a^b	X			X	X		X
a^c	X			X	X		X
b^c	X			X	X	X	X
a^b^c	X			X	X		X

parties gauches (↔)

parties droites (↕)

❸La couverture minimale de l'ensemble de règles (les 33 règles ❷) représente permet de déduire, en utilisant les axiomes d'Armstrong, l'ensemble des règles logiques ❷.

❹Le treillis conceptuel de la relation permet également de déduire l'ensemble de règles logiques, et notamment celle de la couverture minimale. Dans cet exemple, pour déduire la couverture minimale, le raisonnement est le suivant : le descripteur a apparaît dans la description fermée $a \wedge b \wedge c$, par contre, il n'apparaît dans aucune autre description représentée, ce qui signifie que les descriptions a, $a \wedge b$, $a \wedge c$ sont non fermées ; on peut donc déduire que tous les objets décrits par a sont également décrits par b et c, et donc, $a \rightarrow b \wedge c$.

FIG. 5 – Comparaison entre une couverture minimale et un treillis de Galois.

3. toutes les règles d'association (logiques ou non) ainsi que leurs probabilités conditionnelles peuvent être déduites à partir des intentions (représentées) ou des pseudo-intentions (déduites) [Pasquier, 2000].

L'utilisation d'un treillis de Galois pour représenter les règles impose cependant de représenter l'intégralité du treillis –la non représentation d'une description sur le treillis de Galois signifie que celle-ci n'est pas fermée– alors que la représentation d'une partie de la couverture minimale permet tout de même le raisonnement. Dans l'exemple de la figure 5, si seul les arcs $b \rightarrow b \wedge c$ et $c \rightarrow b \wedge c$ sont représentés, il est impossible de déduire la règle logique $a \rightarrow b \wedge c$, en effet, l'information sur la fermeture de a n'est pas représentée.

Une autre limite de la représentation de règles d'association en utilisant un treillis de Galois apparaît lorsqu'il n'existe aucune règle logique sur une relation ; dans ce cas, toutes les descriptions fréquentes pouvant être découvertes sont fermées, il n'existe aucune pseudo-intention et donc le treillis de Galois est équivalent au treillis d'inclusion des descriptions.

4.2.2 Ensembles δ-libres

Pour combler cette dernière limite, Boulicaut et al. ont proposé une nouvelle notion, les descriptions δ-libres [Boulicaut *et al.*, 2000], permettant d'étendre la notion de fermeture pour tenir compte d'inclusion quasi-strictes. Une description est δ-libre si il n'existe aucune règle entre des sous-ensembles de cette description invalidée par au plus δ objets de la base (δ étant supposé petit). L'ensemble des descriptions δ-libres fréquentes permet d'approximer l'ensemble des descriptions, –avec un taux d'erreur borné sur le support et la probabilité conditionnelle– tout en étant moins volumineux et plus rapide à calculer.

5 Extension aux règles d'association

5.1 Le constat

Les axiomes d'Armstrong ne sont pas systèmatiquement applicables dans le cas d'implications non strictes[4] telles que le sont les règles d'association. Un contre exemple classique de l'axiome d'Armstrong de transitivité (29) est l'ensemble des deux implications :

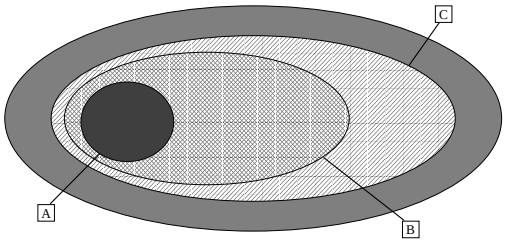
$$\left\{ \begin{array}{l} \texttt{autruche} \rightarrow \texttt{oiseau}, \\ \texttt{oiseau} \rightarrow \texttt{vole} \end{array} \right\}$$

à partir duquel on ne peut bien sûr pas déduire

$$\texttt{autruche} \rightarrow \texttt{vole}.$$

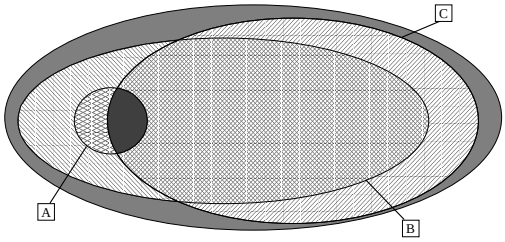
La figure 6 illustre les cas limites de validation et d'invalidation de l'axiome d'Armstrong de transitivité pour l'implication statistique.

[4] pour lequel il existe des exemples invalidant l'implication.

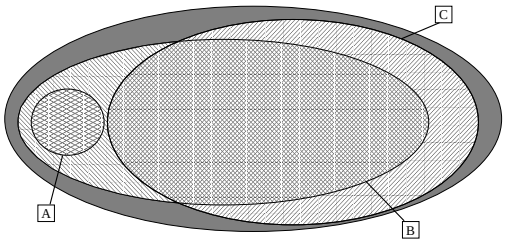


Note : [A] ([B], ...) représente ici l'ensemble des objets pour lesquels a est vrai (b, ..., respectivement)

❶ Les deux implications statistiques $a \rightarrow b$ et $b \rightarrow c$ sont observées, et l'implication statistique $a \rightarrow c$ est également observée
⇒ transitivité !

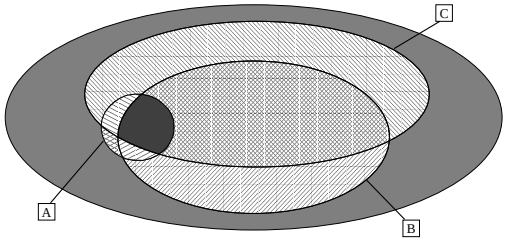


❷ Les deux implications statistiques $a \rightarrow b$ et $b \rightarrow c$ sont observées, mais aucune relation entre a et c n'est observée
⇒ absence de transitivité !

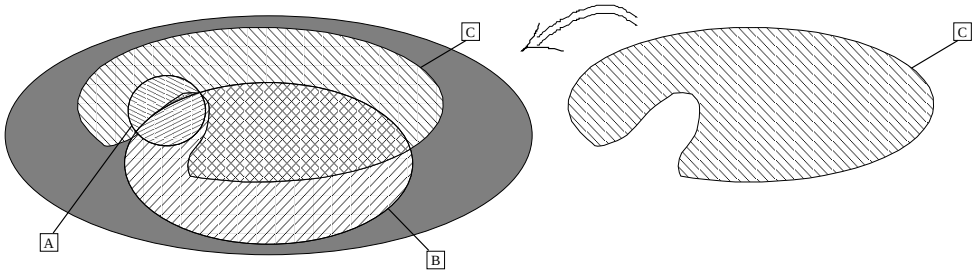


❸ Les deux implications statistiques $a \rightarrow b$ et $b \rightarrow c$ sont observées, mais l'implication statistique $a \rightarrow \neg c$ est observée
⇒ anti-transitivité !

FIG. 6 – Cas limites de validation/invalidation de l'axiome de transitivité pour l'implication statistique.



❶ L'implication statistique $a \rightarrow b$ est observée, l'implication statistique $a \wedge c \rightarrow b \wedge c$ également (l'axiome d'augmentation fonctionne correctement).



❷ L'implication statistique $a \rightarrow b$ est observée, mais ici, les exemples contribuant à $a \wedge c$ contredisent $a \wedge c \rightarrow b$; donc ici, malgré l'observation de $a \rightarrow b$, on n'observe pas $a \wedge c \rightarrow b \wedge c$.

FIG. 7 – Cas limites de validation/invalidation de l'axiome d'augmentation pour l'implication statistique.

D'une manière similaire, des cas d'invalidation de l'axiome d'Armstrong d'augmentation (28) sont observables, par exemple,

$$\texttt{oiseau} \rightarrow \texttt{vole},$$

et pourtant, on peut imaginer ne pas pouvoir observer

$$\texttt{oiseau} \wedge \texttt{lourd} \rightarrow \texttt{vole} \wedge \texttt{lourd}^{5}.$$

La figure 7 illustre quant à elle les cas limites de validation et d'invalidation de l'axiome d'Armstrong d'augmentation pour l'implication statistique.

5.2 Quelques solutions

La limite du modèle proposé dans cet article est donc une limite dûe aux différences de comportement vis à vis des règles de réécritures, entre les règles d'association et les

[5] tout en ayant des objets `lourd`s et qui `vole`nt, l'Airbus *A3xx*, par exemple.

dépendances fonctionnelles.

Il existe des travaux ayant des objectifs similaires, utilisant les propriétés d'autres représentations de connaissances (dépendances ou treillis de Galois, par exemple). Le problème de la mise en oeuvre d'une représentation adéquate en fonction des propriétés inférentielles de la connaissance représentée reste ouverte dans le cas général.

Dans le cas particulier de l'utilisation des propriétés de la logique d'ordre 0 pour les règles d'association, trois grandes catégories de pistes peuvent être envisagées :

1. on fait l'hypothèse que les règles d'association se comportent selon la logique d'ordre 0 ; cette hypothèse peut-être envisagée pour des sous-ensembles de règles, après validation par un expert du domaine, impliqué dans le processus de découverte de connaissances à partir des données, en particulier, si l'exploitation des règles d'association nécessite l'adhésion à des propriétés de la logique d'ordre 0 (ce qui peut être nécessaire pour l'exploitation dans le cadre d'un moteur d'inférences pour un système expert).
2. Le paramétrage de seuils sur les indices de qualité des règles (faible support, confiance élevée) peut permettre d'obtenir un comportement des règles d'association plus proche de celui des règles logiques. Il n'existe cependant pas de formule permettant de déterminer, dans le cas général, un seuil limite sur les indices de qualité usuels, pour un niveau de validité donné (en dehors du cas où un seuil de 100% sur la confiance permet d'obtenir des règles d'association pour lesquelles les inférences logiques sont valides sur l'espace d'apprentissage).
3. L'élimination des redondances, sous l'hypothèse d'un comportement logique des règles peut être envisagée, si elle est associée à une démarche interactive, permettant à l'utilisateur de vérifier ou d'infirmer ses hypothèses -obtenues selon un raisonnement logique- au fur et à mesure de son raisonnement (nous avons testé ce type de méthode interactive, sur un modèle de raisonnement similaire [Kuntz *et al.*, 2000]). Ce type de démarche peut également permettre la mise en évidence et la description des cas d'exceptions au comportement logique rencontrés pour des règles d'association ou des ensembles de règles d'association.

6 Illustration

Afin de montrer l'intérêt de cette méthode, nous recherchons la couverture minimale sur des ensembles de règles produites à partir de données synthétiques et nous mesurons la qualité du résumé exprimé par la couverture minimale de chaque ensemble de règles en évaluant :

- le nombre de règles de la couverture minimale, comparée au nombre de règles d'association découverte,
- dans quelle mesure les règles de la couverture minimale sont des règles d'association valides (existent selon les critères de qualité fixés lors de la découverte de règles d'association),
- dans quelle mesure les règles qu'un utilisateur peut déduire de la couverture minimale sont des règles d'association valides.

Nom	Valeurs	Interprétation
N	$\in \{100, 1000, 10000\}$	nombre d'objets de la base de données.
N_items	10	nombre de descripteurs booléens.
$p(item_i)$	$\in [0.05 : 0.5]$	pour chaque descripteur $item_i$ de chaque objet o_j de la base, la probabilité que $item_i$ soit un descripteur de o_j.
$N\_descripteurs$	10	nombre maximum de descripteurs dans une règle.
$min\_support$	0.1	support minimum des règles produites.
min_conf	$\in \{0.8, 0.9, 1\}$	confiance minimum des règles produites.
min_φ	$\in \{0.9, 0.95, 1\}$	intensité d'implication minimum des règles produites.

TAB. 3 – Paramètres de génération des jeux d'essai.

6.1 Jeux d'essai

Les ensembles de règles utilisés sont construits en appliquant un algorithme de découverte de règles d'association sur des ensembles de données synthétiques, dont les paramètres de production correspondent à des bases de données réelles. La table 3 résume ces paramètres.

70 bases de données synthétiques et autant d'ensembles de 50 à 3078 règles (357 règles en moyenne) ont ainsi été générés.

6.2 Résultats

6.2.1 Limitation du volume de règles

Les couvertures minimales de ces 70 ensembles ont été calculées. Pour ces jeux d'essais, la couverture minimale représente entre 0.3% et 100%, pour une moyenne de 4%, du nombre de règles des ensembles de règles de départ. Ce poucentage diminue beaucoup lorsque le nombre de règles devient plus grand (courbe des + sur la figure 8). En effet, la limite maximale du nombre d'attributs (descripteurs booléens), dans les bases de données synthétique influe directement sur les tailles des couvertures minimales. On peut ici remarquer l'intérêt de la méthode, dans la limitation importante du volume de règles représentées.

6.2.2 Validité des règles de la couverture minimale

Pourtant, en moyenne, 73.7% des règles de la couverture minimale sont des règles d'association valides (correspondant aux critères de qualité utilisés pour la génération de l'ensemble de règles de départ), mais ce pourcentage diminue également lorsque le nombre de règles devient plus grand (courbe des $\times$ sur la figure 8). Cependant, lors de ces expériences, il est toujours resté au delà de 30%.

6.2.3 Confirmation des règles que l'utilisateur peut déduire, au moyen d'indices de qualité

Toutes les règles d'association découvertes peuvent être déduites de la couverture minimale (par définition de la couverture minimale), en appliquant les axiomes d'Armstrong, mais l'application des axiomes d'Armstrong peut amener l'utilisateur à déduire d'autres règles qui, elles, ne sont pas valides. Afin de mesurer la validité des règles, nous les confrontons à un test utilisant des indices de qualité.

Les règles que l'utilisateur peut déduire[6] à partir de la couverture minimale sont des règles d'association valides dans 63.9% des cas en moyenne, pour un minimum de 22.2%, selon les critères de sélection au moyen d'indices de qualité fixés pour la génération des règles à partir des bases de données synthétiques. Ce taux de validité des règles ne semble pas se dégrader lorsque les ensembles de règles deviennent plus gros (courbe des $*$ de la figure 8). Ce taux semble être d'autant plus élevé que le seuil de confiance lors de la production de règles d'association est élevé (premier tableau de la figure 9) ; on aurait pu s'attendre à 100% de validité pour un seuil de confiance à 1, mais les 98.1% obtenus s'expliquent par l'existence de règles calculées en appliquant les axiomes d'Armstrong, mais qui ont une intensité d'implication ou un support inférieur au seuils fixés lors de la génération des règles. Le seuil d'intensité d'implication ne semble pas influencer le taux de validité des règles déduites.

La limite rencontrée -ici, plus d'un tiers des règles déduites ne sont pas valides- est une limite du raisonnement logique sur les règles d'association, indépendante de la qualité des règles de la couverture minimale. Le raisonnement logique est celui recommandé par la représentation que nous avons choisie (la couverture minimale de l'ensemble des règles, selon les axiomes d'Armstrong) ; néanmoins, la fermeture de l'ensemble de règles de départ est la même que la fermeture calculée sur la couverture minimale. En d'autres termes, l'utilisateur, en appliquant les axiomes d'Armstrong sur les règles d'associations découvertes (sans application de la méthode d'élimination des redondances) obtiendra les mêmes règles qu'en appliquant les axiomes d'Armstrong sur la couverture minimale. La dégradation de la qualité des règles d'association doit donc être considérée comme étant due au modèle de raisonnement logique, imposé à l'utilisateur, et non par rapport à la représentation de la couverture minimale elle-même.

[6] Dans le cadre de cette expérimentation, nous faisons une approximation du raisonnement de l'utilisateur en considérant que celui-ci sera capable de déduire toutes les règles se trouvant dans la fermeture de la couverture minimale, en utilisant les axiomes d'Armstrong.

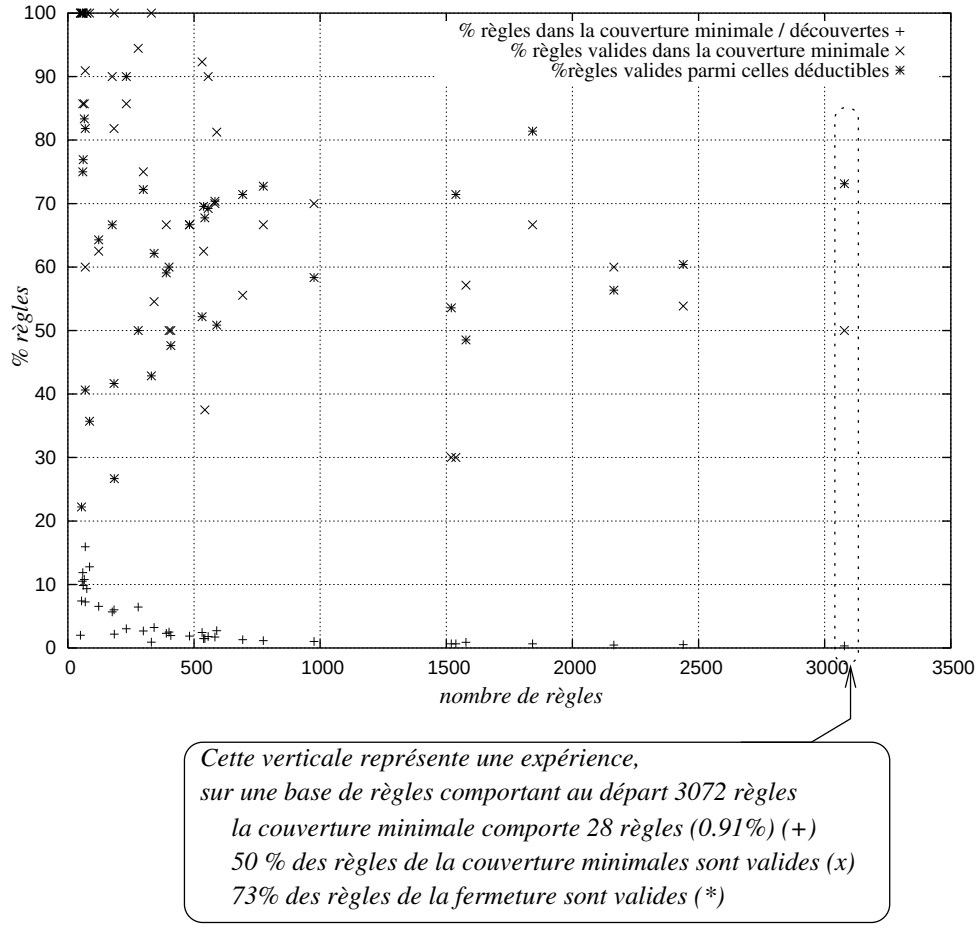


FIG. 8 – Résultats expérimentaux

min_conf	Taux de confirmation par l'indice de confiance, des règles déduites de la couverture minimale.
0.8	66.1%
0.9	70.5%
1	98.1%
$min\_\varphi \in \{0.9, 0.95, 1\}$	

min_φ	Taux de confirmation par l'indice d'intensité d'implication, des règles déduites de la couverture minimale.
0.9	72.4%
0.95	73.3%
1	76%
$min\_conf \in \{0.8, 0.9, 1\}$	

FIG. 9 – Résultats expérimentaux : confirmation des règles déduites à partir de la couverture minimale, en utilisant des tests au moyen des indices de qualité de confiance et d'intensité d'implication

7 Conclusion

Nous avons appliqué à des ensembles de règles d'association une méthode permettant d'éliminer les règles redondantes, c'est-à-dire celles qu'un utilisateur peut lui-même déduire en utilisant des propriétés logiques. Les propriétés des dépendances fonctionnelles fournissent un exemple de règles de réécritures qui sont à la fois valides pour le calcul propositionnel, mais qui sont aussi, dans le cadre de la conception de modèles de bases de données, interprétables et applicables par un utilisateur. Après avoir rappelé le cadre théorique de l'élimination des dépendances fonctionnelles redondantes et décrit l'algorithme de calcul de la couverture minimale d'un ensemble de dépendances fonctionnelles nous avons comparé l'utilisation de ce modèle sur des ensembles de règles d'association à d'autres méthodes permettant de limiter le volume de règles représentées. Nous avons ensuite détaillé les cas de règles d'association, pour lesquels les principes d'interprétation d'ensembles de dépendances fonctionnelles sont invalides. Enfin, nous avons montré, sur des bases de données synthétiques, que cette méthode permettait de réduire très considérablement le nombre de règles exposées à l'utilisateur, tout en lui permettant de déduire lui-même l'ensemble des règles éliminées, au prix d'une certaine approximation : d'une part, il n'existe pas de règles de calcul permettant, dans le cas général, à l'utilisateur de déduire lui même les indices de qualité associés individuellement aux règles masquées, et d'autre part, l'utilisateur peut être amené à déduire des règles invalides, c'est-à-dire n'existant pas dans l'ensemble de règles initial, selon les critères de qualité fixés lors de la génération de ces règles.

Néanmoins, la concision de la représentation obtenue et le recours à des propriétés logiques usuelles et simples à réaliser pour l'utilisateur en font une méthode intéressante, qui peut être utilisée en première approche lors de la découverte de règles d'association, avant, par exemple, une fouille interactive dans les règles, permettant alors à l'utilisateur d'examiner plus précisemment certains motifs de règles. Cette mé-

thode apporte également un point de vue alternatif dans l'évaluation de la qualité des règles : non plus seulement considérer la qualité individuelle de chaque règle, mais aussi la qualité d'un ensemble de règles mises en évidence, dans un processus de prise de décision.

Un programme *PERL*, implémentant les algorithmes décrit dans cet article est téléchargeable à l'adresse
`http ://www.fc.univ-nantes.fr/oasis/EGC/logiciels/redondances/`.

Références

[Agrawal *et al.*, 1996] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, et A. Inkeri Verkamo. Fast discovery of association rules. In Fayyad et al. [1996], pages 307–328.

[Atkins, 1988] J. Atkins. A note on minimal covers. *SIGMOD RECORD*, 17(4) :16–21, December 1988.

[Boulicaut *et al.*, 2000] J.-F. Boulicaut, A. Bykowski, et C. Rigotti. Approximation of frequency queries by means of free-sets. In *Proceedings of Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Computer Science, pages 75–85. Springer-Verlag, 2000.

[Brachman et Anand, 1996] J.R. Brachman et T. Anand. The process of knowledge discovery in databases : a human-centered approach. In Fayyad et al. [1996], pages 37–58.

[Briand *et al.*, 1986] H. Briand, J.B. Crampes, Y. Hebrail, D. Herin Aime, J. Kouloumdjian, et R. Sabatier. *Les systèmes d'information.* éditions DUNOD, 1986.

[Delobel et Adiba, 1982] C. Delobel et M. Adiba. *Bases de données et systèmes relationnels.* DUNOD Informatique, 1982.

[Dumitriu *et al.*, 2000] L. Dumitriu, C. Tudorie, E. Pecheanu, et A. Istrate. A new algorithm for finding association rules. In *Proceedings of Data Mining*, volume 2, pages 195–202. Wessex Institute of Technology, WIT Press, 2000.

[Fayyad *et al.*, 1996] U.M. Fayyad, G. Piatetsky-Sapiro, et P. Smyth, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press, 1996.

[Fleury, 1994] L. Fleury. Adaptation d'une méthode de recherche de la couverture minimale d'un ensemble de dépendances fonctionnelles pour l'élimination des redondances dans un système de règles. INFORSID, Aix en Provence, 1994.

[Fleury, 1996] L. Fleury. *Découverte de connaissances dans une base de données de gestion des ressources humaines.* PhD thesis, Université de Nantes, 1996.

[Ganascia, 1987] J.-G. Ganascia. *AGAPE et CHARADE : deux techniques d'apprentissage symbolique appliquées à la construction de bases de connaissances.* PhD thesis, Université de Paris Sud, 1987.

[Guigues et Duquennes, 1986] J.-L. Guigues et V. Duquennes. Familles minimales d'implications informatives résultant d'un tableau de données binaires. In *Mathématiques et sciences humaines*, number 95, pages 5–18. 1986.

[Hipp *et al.*, 2000] J. Hipp, U. Güntzer, et G. Nakhaeizadeh. Mining association rules : deriving a superior algorithm by analysing today's approaches. In *Proceedings of the 4^{th} European Conference on Principles of Data Mining and Knowledge Discovery*, volume 1910 of *Lecture Notes in Computer Science*, pages 159–168. Springer Verlag, 2000.

[Kaufmann, 1987] A. Kaufmann. *Nouvelle logique pour l'intelligence artificielle.* Mathématiques appliquées. Editions Hermes, 1987.

[Kuntz *et al.*, 2000] P. Kuntz, R. Lehn, et H. Briand. A user-driven process for mining association rules. In *Proceedings of Principles of Data Mining and Knowledge Discovery*, volume 1910 of *Lecture Notes in Computer Science*, pages 483–489. Springer-Verlag, 2000.

[Lehn *et al.*, 1999] R. Lehn, F. Guillet, P. Kuntz, H. Briand, et J. Philippé. Felix : An interactive rule mining interface in a kdd process. In *Proceedings of the 10^{th} Mini-Euro Conference, Human Centered Processes, HCP'99*, pages 169–174. Ecole Nationale Supérieure des Télécommunications de Bretagne, 1999.

[Lopes *et al.*, 1999] S. Lopes, J.-M. Petit, et L. Lakhal. Efficient discovery of functional dependancies and armstrong relations. Rapport de recherche LIMOS, Université Blaise Pascal, Clermont-Ferrand II, 1999.

[Mannila et Räihä, 1992] H. Mannila et K.-J. Räihä. *The Design of Relational Databases.* Addison-Wesley, 1992.

[Pasquier *et al.*, 1999] N. Pasquier, Y. Bastide, R. Taouil, et L. Lakhal. Efficient mining of association rules using closed itemset latices. *Information Systems*, 24(1) :25–46, 1999.

[Pasquier, 2000] N. Pasquier. *Data Mining : Algorithmes d'Extraction et de Réduction des Règles d'Association dans les Bases de Données.* PhD thesis, Université de Clermont-Ferrand II, 2000.

[Ullman, 1982] J.D. Ullman. *Principles of Database Systems.* Computer Science Press, 1982.

[Ullman, 1989a] J.D. Ullman. *Principles of Database and Knowledge–base Systems*, volume 1. Computer Science Press, 1989.

[Ullman, 1989b] J.D. Ullman. *Reasoning about functional dependencies*, chapter 7.3, pages 382–392. Volume 1 of [1989a], 1989.