Enseignement de l'informatique

POLYCOPIE

Optimisation sous Contraintes et Fouille de données : partie 2

LEBBAH Yahia
Chargé de cours, Université d'Oran 1

Département Informatique, Faculté FSEA, Université d'Oran 1
B.P. 1524, El-M'Naouar Oran, Algérie

Version du 5 novembre 2022

Table des matières

Chapitre 1

Fouille des motifs ensemblistes (itemset)

[1] Cette section présente les principales notions liées à l'extraction de motifs fréquents.

1.1 Contexte et définitions - Motifs fréquents

— Soit $\mathcal{I} = \{I_1, ..., I_n\}$ un ensemble de littéraux distincts appelés **items** et $\mathcal{T} = \{1, ..., m\}$ un ensemble d'identifiants de **transactions**.
— Un motif ensembliste d'items est un sous-ensemble non vide de $\mathcal{I}$. Ces motifs sont regroupés dans le langage $\mathcal{L}_\mathcal{I} = 2^\mathcal{I} \backslash \emptyset$.
— Un jeu de données transactionnel est l'ensemble $\mathbf{r} \subseteq \mathcal{I} \times \mathcal{T}$.

Définition 1 (Couverture et fréquence) *La couverture d'un motif x est l'ensemble de tous les identifiants de transactions qui le supportent : $couv_\mathcal{T}(x) = \{t \in \mathcal{T} | \forall i \in x, (i,t) \in r\}$. La fréquence d'un motif x représente le cardinal de sa couverture $freq_\mathcal{T}(x) = |couv_\mathcal{T}(x)|$.*

L'extraction de motifs sous contraintes consiste à extraire les motifs satisfaisant une contrainte C à partir d'un jeu de données $\mathbf{r}$. La contrainte de **motifs fréquents** est définie en sélectionnant les motifs dont la fréquence est supérieure à un seuil donné min_{fr}. La contrainte de *taille* contraint le nombre d'items d'un motif x.

1. Des extraits sont pris de [Maamar et al., 2015]

transaction database

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{b, c, e\}$
10: $\{a, d, e\}$

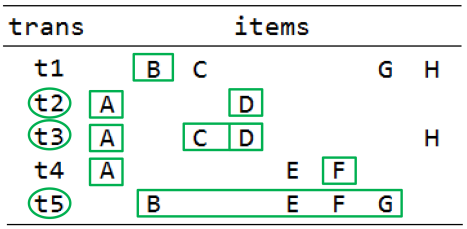
frequent item sets

0 items	1 item	2 items	3 items
$\emptyset$: 10	$\{a\}$: 7	$\{a, c\}$: 4	$\{a, c, d\}$: 3
	$\{b\}$: 3	$\{a, d\}$: 5	$\{a, c, e\}$: 3
	$\{c\}$: 7	$\{a, e\}$: 6	$\{a, d, e\}$: 4
	$\{d\}$: 6	$\{b, c\}$: 3	
	$\{e\}$: 7	$\{c, d\}$: 4	
		$\{c, e\}$: 4	
		$\{d, e\}$: 4	

- In this example, the minimum support is $s_{\min} = 3$ or $\sigma_{\min} = 0.3 = 30\%$.
- There are $2^5 = 32$ possible item sets over $B = \{a, b, c, d, e\}$.
- There are 16 frequent item sets (but only 10 transactions).

FIGURE 1.1 – Exemple d'une fouille dans une table transactionnelle (Image extraite de [Bourgelt, 2015]))

Une limite bien connue de l'extraction de motifs est le nombre de motifs produits qui peut être très grand. Les *représentations condensées* de motifs satisfaisant une contrainte C ont été introduites pour augmenter la rapidité d'exécution des algorithmes d'extraction de motifs et réduire le nombre de motifs obtenus. Pour les motifs ensemblistes et la contrainte de fréquence minimale, la représentation la plus usuelle est celle fondée sur les *motifs fermés*.



	A	B	C	D	E	F	G	H
t1	0	1	1	0	0	0	1	1
t2	1	0	0	1	0	0	0	0
t3	1	0	1	0	0	0	0	1
t4	1	0	0	0	1	1	0	0
t5	0	1	0	0	1	1	1	0

FIGURE 1.2 – Exemple d'une table transactionnelle

Définition 2 (Motif fermé) *Un motif $x \in \mathcal{L}_{\mathcal{I}}$ est* **fermé** *par rapport à la fréquence ssi $\forall y \supsetneq x, freq(y) < freq(x)$.*

Les motifs fermés structurent le treillis des motifs en *classes d'équivalence* (pour plus de détail voir le chapitre **??**). Tous les motifs d'une même classe d'équivalence ont la même couverture. Les motifs fermés correspondent aux éléments maximaux (au sens de la taille des motifs) des classes d'équivalence.

Par ailleurs, l'utilisateur est souvent intéressé par la découverte de motifs plus riches satisfaisant des contraintes portant sur un ensemble de motifs et non plus un seul motif. Ces contraintes sont définies comme étant des contraintes sur des *ensembles de motifs* [De Raedt and Zimmermann, 2007] ou sur des motifs *n-aires* [Khiari et al., 2010] (voir le chapitre 2). Une approche intéressante consiste à traiter ce type de motifs via les top-$k$motifs.

Définition 3 (top-$k$motifs) *Soit m une mesure d'intérêt et k un entier. Les top-$k$motifs est l'ensemble des k meilleurs motifs par rapport à la mesure m :*

$$\{x \in \mathcal{L}_{\mathcal{I}} \mid \quad freq_{\mathcal{T}}(x) \geq 1 \wedge \quad \nexists y_1, \ldots, y_k \in \mathcal{L}_{\mathcal{I}} : \ \forall 1 \leq j \leq k, m(y_j) > m(x)\}$$

Exemple : interpréter les top-$k$motifs dans la table transactionnelle de la Figure 1.2.

1.2 Algorithmes de recherche des motifs ensemblistes fréquents

On peut naïvement penser à un algorithme exhaustif qui parcourt tous les sous-ensembles d'items et teste s'ils respectent la contrainte de fréquence (et autres contraintes). Bien évidemment, un tel algorithme serait très couteux et exponentiel. Pour éviter un tel parcours, nous ferons appel à la propriété suivante :

$$\forall I \subseteq \mathcal{I}, \forall J \supseteq I, freq_{\mathcal{T}}(J) \leq freq_{\mathcal{T}}(I).$$

Le principe : si un itemset est étendu, sa fréquence décroit.

Cette propriété est dite qualifié de "propriété Apriori", en référence à l'algorithme Apriori [Agrawal and Srikant, 1994]. Elle est aussi qualifié de propriété d'anti-monotonie (voir figures ci-dessous).

La contraposé est aussi vraie : tous les sous-ensemble d'un itemset fréquent sont fréquents.

Principes de recherche des itemsets fréquents :

- La procédure de recherche standard consiste en une approche d'énumération qui énumère les candidats et vérifie la satisfaction de la contrainte de fréquence.
- La procédure améliore l'approche naïve en exploitant la propriété d'anti-monotonie qui évite d'explorer les itemsets dont un de ses sous-ensembles n'est pas fréquent.
- L'espace de recherche est l'ensemble muni d'un ordre partiel $(2^{|\mathcal{I}|}, \subseteq)$.
- La structure d'ordre partiel permet d'éviter des candidats en exploitant la propriété d'anti-monotonie. L'approche est donc une approche descendante qui démarre du candidat vide, l'étend au fur et à mesure. On se prive de continuer toute branche descendante d'un candidat non-fréquent.

transaction database

1: $\{a, d, e\}$
2: $\{b, c, d\}$
3: $\{a, c, e\}$
4: $\{a, c, d, e\}$
5: $\{a, e\}$
6: $\{a, c, d\}$
7: $\{b, c\}$
8: $\{a, c, d, e\}$
9: $\{b, c, e\}$
10: $\{a, d, e\}$

Blue boxes are frequent item sets, white boxes infrequent item sets.

Hasse diagram with frequent item sets ($s_{\min} = 3$):

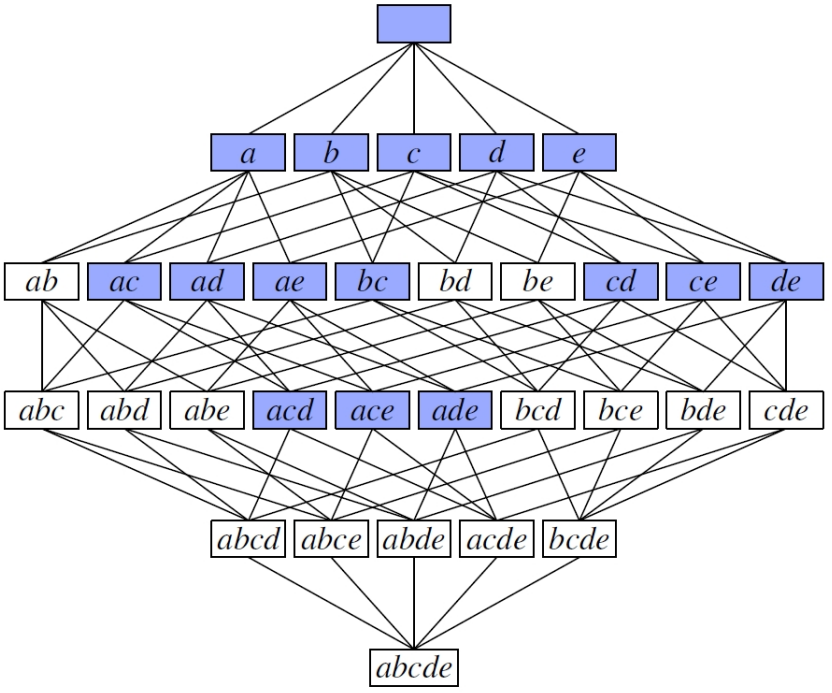


FIGURE 1.3 – Recherche de motifs ensemblistes en exploitant l'anti-monotonie (image de [Bourgelt, 2015])

1.3 Algorithme Apriori

Algorithm: Apriori. Find frequent itemsets using an iterative level-wise approach based on candidate generation.

Input:

- D, a database of transactions;
- *min_sup*, the minimum support count threshold.

Output: L, frequent itemsets in D.

Method:

```
(1)   L1 = find_frequent_1-itemsets(D);
(2)   for (k = 2; Lk−1 ≠ φ; k++) {
(3)       Ck = apriori_gen(Lk−1);
(4)       for each transaction t ∈ D { // scan D for counts
(5)           Ct = subset(Ck, t); // get the subsets of t that are candidates
(6)           for each candidate c ∈ Ct
(7)               c.count++;
(8)       }
(9)       Lk = {c ∈ Ck|c.count ≥ min_sup}
(10)  }
(11)  return L = ∪k Lk;

procedure apriori_gen(Lk−1:frequent (k − 1)-itemsets)
(1)   for each itemset l1 ∈ Lk−1
(2)       for each itemset l2 ∈ Lk−1
(3)           if (l1[1] = l2[1]) ∧ (l1[2] = l2[2])
                  ∧... ∧ (l1[k − 2] = l2[k − 2]) ∧ (l1[k − 1] < l2[k − 1]) then {
(4)               c = l1 ⋈ l2; // join step: generate candidates
(5)               if has_infrequent_subset(c, Lk−1) then
(6)                   delete c; // prune step: remove unfruitful candidate
(7)               else add c to Ck;
(8)           }
(9)   return Ck;

procedure has_infrequent_subset(c: candidate k-itemset;
          Lk−1: frequent (k − 1)-itemsets); // use prior knowledge
(1)   for each (k − 1)-subset s of c
(2)       if s ∉ Lk−1 then
(3)           return TRUE;
(4)   return FALSE;
```

FIGURE 1.4 – Algorithme Apriori (image de [Han, 2005])

Transactional Data for an *AllElectronics* Branch

TID	List of item_IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

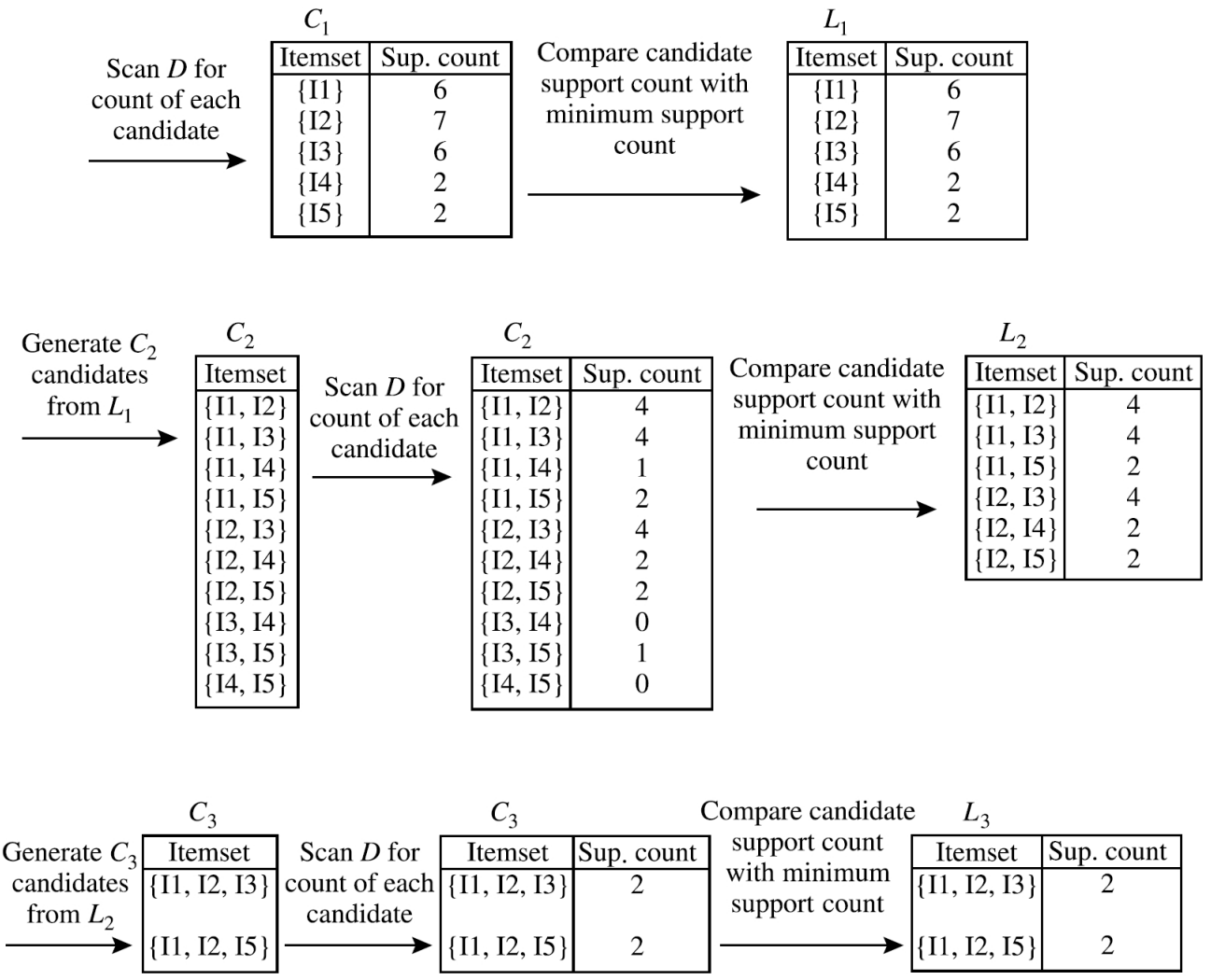


FIGURE 1.5 – Déroulement de l'algorithme Apriori (image de [Han, 2005])

Chapitre 2

Fouille de données déclarative

2.1 Extraction de motifs ensemblistes sous-contraintes

Soit **r** un jeu de données ou $\mathcal{I}$ est l'ensemble de ses n items et $\mathcal{T}$ l'ensemble de ses m transactions. **r** peut être représenté par une matrice booléenne $\mathbf{d} = (d_{t,i})_{t\in\mathcal{T},i\in\mathcal{I}}$, tel que $\forall t \in \mathcal{T}, \forall i \in \mathcal{I}, (d_{t,i} = 1) \Leftrightarrow (i \in t)$.

Variables

- **Soit M le motif recherché. M est représenté par n variables booléennes $\{X_1, X_2, ..., X_n\}$ tel que : $\forall i \in \mathcal{I},\ (X_i = 1)\ ssi\ (i \in M)$.**
- **Le support du motif recherché M est représenté par m variables booléennes $\{T_1, T_2, \ldots, T_m\}$ tel que : $\forall t \in \mathcal{T},\ (T_t = 1)\ ssi\ (M \subseteq t)$.**

Contraintes La relation entre le motif recherché M, son support et le jeu de données **r** est établie par des contraintes réifiées imposant que, pour chaque transaction t, $(T_t = 1)$ ssi M est un sous ensemble de t [Guns et al., 2011] :

$$\forall \mathbf{t} \in \mathcal{T}, (\mathbf{T_t = 1}) \Leftrightarrow \sum_{\mathbf{i}\in\mathcal{I}} \mathbf{X_i} \times (\mathbf{1 - d_{t,i}}) = \mathbf{0} \tag{2.1}$$

L'encodage booléen permet d'exprimer de façon simple certaines mesures usuelles : $freq(M) = \sum_{t\in\mathcal{T}} T_t$ et $taille(M) = \sum_{i\in\mathcal{I}} X_i$. La contrainte de fréquence minimale $freq(M) \geq min_{fr}$ (ou min_{fr} est un seuil) est encodée par la contrainte $\sum_{t\in\mathcal{T}} T_t \geq min_{fr}$. De la même manière, la contrainte de taille minimale $taille(M) \geq \alpha$ (ou α est un seuil) est encodée par la contrainte $\sum_{i\in\mathcal{I}} X_i \geq \alpha$.

Finalement, **la contrainte de fermeture** $fermé_m(M)$ (Avec $m = freq$) est encodée par l'équation (2.2).

$$\forall \mathbf{i} \in \mathcal{I}, (\mathbf{X_i = 1}) \ \Leftrightarrow\ \sum_{\mathbf{t}\in\mathcal{T}} \mathbf{T_t} \times (\mathbf{1 - d_{t,i}}) = \mathbf{0}. \tag{2.2}$$

Interprétation de la formule de fermeture : **L'item i est dans le motif SSI l'item i est dans les transactions actives**.

=> Si un item est dans un motif, il doit être dans les transactions actives. On force à selectionner toutes les transactions couvertes par le motif.

$<=$: Tous les items qui sont dans les transactions actives, ils doivent aussi être dans le motif. On force à garder tous les items couverts par les transactions actives, dans le motif.

Lazaar et al. [Lazaar et al., 2016] ont proposé une seule contrainte, la contrainte ClosedPattern, qui permet de prendre en compte toutes les contraintes de fréquence et de clôture. Nous l'introduisons brièvement ci-dessous.

Soit P le motif inconnu. Le motif P est encodé avec les variables booléennes $P_1, ..., P_n$. Nous dénotons σ l'instanciation partielle des variables $P_1, ..., P_n$ qui ont un domaine avec une seule valeur. Nous dénotons les trois ensembles suivants :

— items présents : $\sigma^+ = \{j \in 1..n \mid P_j = 1\}$,
— items absents : $\sigma^- = \{j \in 1..n \mid P_j = 0\}$,
— items autres : $\sigma^* = \{1..n\} \setminus (\sigma^+ \cup \sigma^-)$.

σ^* est l'ensemble des items libres (variables non instanciées). Si $\sigma^* = \emptyset$ alors σ est une affectation complète.

La contrainte globale ClosedPatterngarantit la contrainte de fréquence et de cloture.

Définition 4 (contrainte globale ClosedPattern) *Soient $P_1, \ldots, P_n$ des variables binaires. Soit SDB la base transactionnelle et θ le support minimal. Etant donné l'affectation complète σ on $P_1, \ldots, P_n$,* $\text{ClosedPattern}_{SDB,\theta}(\sigma)$ *est valide si et seulement si $freq_{SDB}(\sigma^+) \geq \theta$ et σ^+ est clos.*

Bibliographie

[Agrawal and Srikant, 1994] Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile*, pages 487–499. Morgan Kaufmann.

[Bourgelt, 2015] Bourgelt, C. (2015). Frequent pattern mining. Technical report, Support de cours, European Center for Soft Computing, http://www.borgelt.net/teaching.html.

[De Raedt and Zimmermann, 2007] De Raedt, L. and Zimmermann, A. (2007). Constraint-based pattern set mining. In *Proceedings of the Seventh SIAM International Conference on DM*. SIAM.

[Guns et al., 2011] Guns, T., Nijssen, S., and De Raedt, L. (2011). Itemset mining : A constraint programming perspective. *Artificial Intelligence*, 175(12) :1951–1983.

[Han, 2005] Han, J. (2005). *Data Mining : Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Khiari et al., 2010] Khiari, M., Boizumault, P., and Crémilleux, B. (2010). Constraint programming for mining n-ary patterns. In *CP'10*, volume 6308 of *LNCS*, pages 552–567. Springer.

[Lazaar et al., 2016] Lazaar, N., Lebbah, Y., Loudni, S., Maamar, M., Lemière, V., Bessiere, C., and Boizumault, P. (2016). A global constraint for closed frequent pattern mining. In *Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings*, volume 9892 of *Lecture Notes in Computer Science*, pages 333–349. Springer.

[Maamar et al., 2015] Maamar, M., Lazaar, N., Loudni, S., and Lebbah, Y. (2015). Localisation de fautes à l'aide de la fouille de données sous contraintes. In *COSI'2015*.