Découverte de sous-groupes à partir de données séquentielles par échantillonnage et optimisation locale

Romain Mathonat*,**, Jean-François Boulicaut*
Mehdi Kaytoue*,***

*Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France
**Atos, 34 Rue de la Soie, 69100 Villeurbanne
***Infologic, 99 avenue de Lyon, 26500 Bourg-Lès-Valence, France
prenom.nom@insa-lyon.fr

Résumé. La découverte de règles caractéristiques d'une classe reste un problème difficile, particulièrement dans le cadre des données séquentielles (séquences d'ensembles). La découverte de sous-groupes est une bonne formalisation de cette tâche et de nombreux algorithmes dédiés ont été proposés ces 20 dernières années. Une exploration dite exhaustive est souvent inapplicable au vu de la taille de l'espace de recherche, et les méthodes heuristiques de référence, principalement les recherches en faisceau, posent des problèmes de paramétrage. Nous proposons une méthode d'échantillonnage depuis l'espace des motifs pour la découverte de sous-groupes dans des données séquentielles étiquetées. Celle-ci permet, entre autres, de trouver des optima locaux, ne nécessite pas de paramétrage, est indépendante de la mesure de qualité utilisée, et est simple à mettre en oeuvre. La validation empirique sur divers jeux de données nous permet de valider les qualités de cette approche.

1 Introduction

Les données séquentielles sont présentes dans de nombreux contextes applicatifs (analyses de textes ou de vidéos, exploitation de traces d'interactions, supervision de processus industriels, exploration de données en biologie moléculaire, etc). Le cas d'utilisation qui motive nos propres travaux est celui de la supervision industrielle, où les suites d'états du système étudié constituent une séquence. Nous voulons fouiller ces collections de séquences étiquetées par des experts métiers pour découvrir des règles sur les co-occurences de pannes. Ceci permettrait, d'une part, de mieux comprendre le système étudié, d'autre part, de pouvoir construire un moteur de règles explicables, offrant alors des perspectives de prédiction et d'anticipations de certains dysfonctionnements. Une formalisation simple de ce contexte est de considérer qu'il s'agit de découvrir des motifs séquentiels co-occurrents à une variable cible (étiquette ou classe). La découverte de règles caractérisant une classe ou une étiquette a été très étudiée (Novak et al. (2009)), notamment dans le cadre de la découverte de sous-groupes (Wrobel (1997)). La découverte de sous-groupes dans des données étiquetées consiste à trouver des motifs (e.g., des motifs séquentiels), également appelés descriptions, qui définissent en intention des objets

(e.g., des séquences) pour lesquelles une mesure de qualité indique qu'ils se répartissent d'une façon particulière entre les classes ou étiquettes. Nous décidons de travailler à l'exploitation de collections de séquences d'itemsets étiquetées. Pour la découverte de sous-groupes, les approches classiques d'énumération exhaustives comme celle de SD-MAP (Atzmüller et Puppe (2006)) posent rapidement un problème de passage à l'échelle. Dans le cas simple des itemsets, deux facteurs influent sur la faisabilité du calcul : le nombre d'items et le nombre de transactions dans la base de données. Concernant notre type de données séquentielles, nous avons comme facteur supplémentaire le nombre d'itemsets pour chaque séquence. De fait, l'espace de recherche devient rapidement trop grand pour pouvoir être exploré exhaustivement et il faut se contenter d'approches heuristiques qui ne peuvent détecter que certains des sous-groupes intéressants. De plus, les meilleurs motifs (et donc sous-groupes) ne doivent pas être *redondants* : fournir des motifs très similaires à un expert limite la confiance dans la méthode et freine la dissémination de tels outils. Diverses approches heuristiques ont été proposées mais partagent les deux problèmes qui sont le besoin de paramétrisation et la redondance des résultats obtenus. Récemment, plusieurs approches heuristiques prometteuses d'échantillonnage dans l'espace des motifs ont été étudiées (Boley et al. (2011); Egho et al. (2017); Diop et al. (2018)). Il y a eu notamment le travail sur l'algorithme `Misère` (Egho et al. (2017)) qui propose la détection par échantillonnage de sous-groupes dans des séquences d'items et donc un contexte de données séquentielles simplifié. Les auteurs montrent l'intérêt qu'il y a dans un échantillonnage qui se construit à partir de l'une des séquences présentes dans les données. Nous proposons une exploration de l'espace de recherche pour des séquences d'itemsets qui développe certaines des propositions présentes dans `Misère` pour l'échantillonnage de motifs suivi d'un processus d'exploration locale et d'extraction d'optima locaux. Notre méthode présente plusieurs avantages hérités de `Misère` : elle donne des résultats à tout moment de l'exécution, bénéficie de la recherche aléatoire pour limiter la redondance des résultats, et est indépendante de la mesure de qualité utilisée. Nous l'améliorons en étendant le langage de motifs (séquence d'itemsets), et en permettant la découverte d'optima locaux, sans paramétrage de l'utilisateur. D'un point de vue technique, nous utilisons une représentation verticale des données sous la forme de bitsets pour l'amélioration des performances et avons porté une attention particulière à la consommation mémoire. Notons enfin que la méthode est facilement parallélisable. La suite de cet article est organisée comme suit : la Section 2 introduit formellement le problème, la Section 3 discute l'état de l'art, la Section 4 détaille notre proposition qui est ensuite validée empiriquement dans la Section 5.

2 Formalisation du problème traité

Soit $\mathcal{I}$ un ensemble d'*éléments*. Tout sous-ensemble $X \subseteq \mathcal{I}$ est appelé un *itemset*. Une *séquence* $s = \langle X_1, ..., X_n \rangle$ est une liste ordonnée de $n > 0$ itemsets. n est la longueur de la séquence, et $\sum_{i=1}^{n} |X_i|$ est sa taille. Une base de données $\mathcal{D}$ est un ensemble de $|\mathcal{D}|$ séquences d'itemsets. Les séquences peuvent avoir diverses longueurs et tailles et sont identifiées de manière unique (voir Table 1). Chaque séquence comprend une étiquette c parmi un ensemble d'étiquettes C. On note $\mathcal{D}_c \subseteq \mathcal{D}$ l'ensemble des séquences ayant l'étiquette c.

Sous-séquence Une séquence $s = \langle X_1, ..., X_{n_s} \rangle$ est une *sous-séquence* d'une séquence $s' = \langle X'_1, ..., X'_{n'_s} \rangle$, noté $s \sqsubseteq s'$, s.s.i il existe $1 \leq j_1 < ... < j_{n_s} \leq n'_s$ tel que $X_1 \subseteq X'_{j_1}, ..., X_{n_s} \subseteq$

id	$s \in \mathcal{D}$	$class(s)$
s_1	$\langle\{a\},\{abc\},\{ac\},\{d\},\{cf\}\rangle$	$+$
s_2	$\langle\{ad\},\{c\},\{bc\},\{ae\}\rangle$	$+$
s_3	$\langle\{ef\},\{ab\},\{df\},\{c\},\{b\}\rangle$	$-$
s_4	$\langle\{e\},\{g\},\{abf\},\{c\},\{b\},\{c\}\rangle$	$-$

TAB. 1: Un exemple de base de données $\mathcal{D}$.

$X'_{j_{n_s}}$. Dans Table 1, $\langle\{a\},\{bc\}\rangle$ est une sous-séquence de s1 et de s2.

Dans les définitions qui suivent, nous omettrons l'ajout du paramètre base de données $\mathcal{D}$ lorsqu'il n'y a pas d'ambiguïté, afin de simplifier les notations.

Définition en extension, support et fréquence La *définition en extension* ou *extension* d'un motif séquentiel s est $ext(s) = \{s' \in \mathcal{D} \mid s \sqsubseteq s'\}$. Le *support* d'un motif séquentiel s est $supp(s) = |ext(s)|$. Sa *fréquence* est $freq(s) = supp(s)/|\mathcal{D}|$. L'extension de $\langle\{a\},\{bc\}\rangle$ dans les données de Table 1 est $\{$s1, s2$\}$.

Mesure de qualité Soit $\mathcal{S}$ l'ensemble des sous-séquences possibles. Une mesure de qualité φ est une fonction $\mathcal{S} \to \mathbb{R}$. La précision, définit par $\frac{supp(s,\mathcal{D}_c)}{supp(s,\mathcal{D})}$, est une mesure de qualité.

S-extension, I-extension Une séquence s_b est une S-extension de $s_a =< X_1, X_2, ..., X_n >$ par un item x si $\exists i, 0 \leq i \leq n$ tel que $s_b =< X_1, ..., \{x\}_i, ..., X_n >$. Une séquence s_b est une I-extension de $s_a =< X_1, X_2, ..., X_n >$ par un item x si $\exists 1 \leq i \leq n$ tel que $s_b =< X_1, ..., X_i \cup \{x\}, ..., X_n >$. Par exemple, $\langle\{ab\},\{b\}\rangle$ est une I-extension de $\langle\{a\},\{b\}\rangle$

Voisinage direct vertical ou horizontal On appelle *voisinage direct vertical* d'un motif s l'ensemble des motifs identiques à s si (1) on applique à s une I ou S-extension et (2) on applique à s une suppression d'un item. On appelle *voisinage direct horizontal* d'un motif s l'ensemble des motifs identiques à s si l'on modifie un item de s par un autre item du jeu de données.

Optimum local Soit $N(s)$ le voisinage direct d'un motif s. $r^\star$ est un optimum local de $\mathcal{S}$ par une mesure de qualité φ s.s.i $\forall r \in N(r^\star), \varphi(r^\star) \geq \varphi(r)$.

Motif non θ-redondant Un ensemble de motifs est non θ-redondant si $\forall s1, s2 \in \mathcal{S} \times \mathcal{S}, sim(s1, s2) \leq \theta$. Notre mesure de similarité est l'indice de Jaccard défini par :

$$sim(s1, s2) = \frac{|ext(s1) \cap ext(s2)|}{|ext(s1) \cup ext(s2)|}$$

Découverte de sous-groupes diversifiés Soit une base de données $\mathcal{D}$, un entier k, une mesure de similarité sim. La découverte de sous-groupes diversifiés a pour but l'extraction de l'ensemble des meilleurs motifs non θ-redondants de taille inférieure ou égale à k, au regard d'une mesure de qualité φ et d'une classe cible c.

3 État de l'art

Les approches classiques d'énumération exhaustive de motifs séquentiels de type SPADE (Zaki (2001)) peuvent être adaptées à notre problème, comme, par exemple (Zhou et al. (2016)). Cependant, la taille de l'espace de recherche devient rapidement trop grande et une exploration exhaustive est souvent inapplicable en pratique.

Les classifieurs de données séquentielles comme, par exemple (Dafé et al. (2015)), pourraient à première vue nous intéresser. Cependant, ils résolvent le problème de la classification d'une donnée nouvelle, et non pas celui de la découverte d'un motif caractéristique de classe. Sur les méthodes d'échantillonnage dans l'espace des motifs, les auteurs de (Diop et al. (2018)) proposent une technique qui garantit que la probabilité de tirage d'un motif est proportionnelle à sa fréquence. Cette méthode est intrinsèquement liée à la recherche de motifs fréquents, et ne peux donc pas être facilement adaptée à notre problème. Notons aussi que les approches d'échantillonnage visant à limiter la taille du jeu de données, par exemple Toivonen (1996), traitent un problème bien différent de celui de l'échantillonnage dans l'espace des motifs.

L'approche de référence dans le cas de la découverte de sous-groupes mais aussi celle de motifs d'exception est la recherche en faisceaux ("Beam Search") décrite par exemple dans (Duivesteijn et al. (2016)). L'idée est d'explorer l'espace de recherche étage par étage, en gardant les motifs les plus intéressants, qui généreront les motifs potentiellement intéressants de l'étage suivant. Cette méthode gloutonne présente le désavantage de devoir ajuster un paramètre en fonction du jeu de données (la taille du faisceau). De plus les motifs trouvés peuvent être redondants. Une approche visant à résoudre le problème de découverte de règles de classifications a été proposée dans Egho et al. (2017). Une mesure de qualité ("level") a été proposée. L'algorithme associé, baptisé `misère`, est basé sur de l'échantillonnage de motifs et la généralisation des éléments tirés afin de proposer de nouveaux motifs de classification. La force de `misère` réside dans le fait qu'un objet du jeu de données est tiré et le motif généré possède donc un support non nul. Cette contribution est perfectible : la séquence tirée aléatoirement dans le jeu de données ne contient pas forcément l'étiquette cible, et `misère` effectue simplement un nouveau tirage à chaque itération, sans garantie de tomber sur un optimum local. Notons enfin que `misère` traite des séquences d'items plutôt que des séquences d'itemsets. Dans Bosc et al. (2018), les auteurs ont proposé une exploration de l'espace de recherche basée sur la recherche arborescente de Monte Carlo dans le cadre des itemsets étiquetées. Cette méthode est donc également basée sur de l'échantillonnage, où chaque tirage renvoie de l'information sur l'espace de recherche. Ces suites d'informations permettent de guider la recherche selon un compromis d'exploitation/exploration. Cette méthode présente cependant l'inconvénient de devoir stocker tous les éléments rencontrés (construction de l'arbre), ce qui peut présenter un budget mémoire élevé. De plus, son adaptation aux données séquentielles est difficile.

Notre algorithme permet à la fois d'extraire des règles de co-occurrences motif-classe (en maximisant une mesure de qualité bien connue appelée $WRAcc$, Lavrac et al. (1999)). Cependant, notre méthode est assez générique pour permettre l'utilisation de n'importe quelle

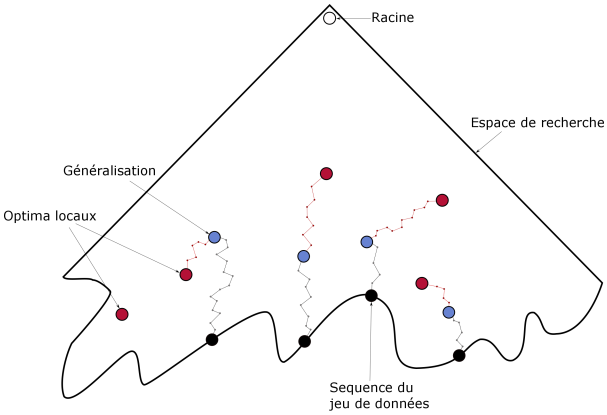

FIG. 1: Fonctionnement de l'algorithme.

mesure, sans exiger de propriétés particulières, comme la monotonie du support. Aucun paramétrage n'est nécessaire. Contrairement à `misère`, nous avons la garantie de trouver des optima locaux grâce au hill-climbing.

4 Échantillonnage de données et optimisation locale

Notre algorithme d'exploration de l'espace de recherche présente l'avantage d'être simple mais suffisamment efficace. A chacune de ses étapes, des variations sont possibles et cette généricité devrait permettre de l'adapter à d'autres problèmes. Une représentation visuelle du fonctionnement de notre algorithme est dans Figure 1 et son pseudo-code est décrit par Algorithme 1. Dans le contexte des motifs séquentiels, l'espace de recherche est a priori infini. Nous avons cependant matérialisé la limite de l'espace de recherche (la frontière du bas sur le schéma) comme étant les derniers motifs dont toutes les I-extensions et S-extensions produiront des motifs de support nul. On peut prouver facilement que tout élément de cette frontière correspond en fait à au moins un élément du jeu de données. La forme de l'espace de recherche est donc totalement dépendante du jeu de données.

4.1 Mesure de qualité

Pour extraire des règles de co-occurences motifs-classe pertinentes, nous choisissons d'utiliser la Weighted Relative Accuracy ($WRAcc$, Lavrac et al. (1999)). Elle permet de comparer la proportion d'éléments possédant l'étiquette ciblée dans le sous-groupe à la proportion d'éléments possédant l'étiquette ciblée dans la population totale.

Weighted Relative Accuracy Soit $c \in C$ une étiquette et s un motif séquentiel :

$$WRAcc(s,c) = freq(s) \times \left(\frac{supp(s, \mathcal{D}_c)}{supp(s, \mathcal{D})} - \frac{|\mathcal{D}_c|}{|\mathcal{D}|} \right)$$

Algorithm 1 Recherche de sous-séquences pertinentes

```
1: fonction TOP-KSOUSSEQUENCES(donnees, k, classeCible)
2:     motifsTrie ← filePriorite()
3:     tant que Budget disponible faire
4:         sequenceActuelle ← tirage(donnees)
5:         sequenceActuelle ← generalisation(sequenceActuelle)
6:         qualiteActuelle ← calculQualite(sequenceActuelle)
7:         faire
8:             variations ← calculVoisinage(sequence)
9:             sequence, qualite = max(variations)
10:            ameliore ← Faux
11:            si qualite > qualiteActuelle alors
12:                qualiteActuelle, sequenceActuelle ← qualite, sequence
13:                ameliore ← Vrai
14:            fin si
15:        tant que ameliore
16:        motifsTrie.ajouteMotif(qualiteActuelle, sequenceActuelle)
17:    fin tant que
18:    retourne motifsTrie.extraireTopKNonRedondant(k)
19: fin fonction
```

Autrement dit, la $WRAcc$ est la différence de précision entre les règles $s \rightarrow c$ et $\langle\rangle \rightarrow c$. Cette différence est pondérée par la fréquence du motif pour éviter l'extraction de sous-groupes peu fréquents et trop précis : trouver une différence de précision parfaite pour des sous-groupes très spécifiques et donc couvrant très peu de données est à éviter. La $WRAcc$ prend ses valeurs dans l'intervalle [-0.25, 0.25].

4.2 Tirage

Dans un premier temps, une séquence est tirée aléatoirement parmi les séquences du jeu de données ayant l'étiquette cible. Ceci permet de s'assurer que le motif généré par la suite contiendra au minimum un élément du jeu de donnée et le motif aura donc un support non nul.

Plusieurs stratégies sont possibles :

— Tirer uniformément un élément du jeu de données avec remise. C'est la stratégie la plus simple que nous utilisons aujourd'hui.

— Tirer uniformément sans remise : ceci permet de ne pas réutiliser un motif déjà utilisé afin de favoriser la diversité, dans une certaine mesure.

— Tirer avec remise, avec diminution de la probabilité de prendre un élément déjà choisi.

4.3 Généralisation

Dans un second temps, cette séquence est généralisée, c'est à dire qu'un nouveau motif plus général est construit à partir de cette séquence. Pour cela, chaque item du motif est examiné, et a une probabilité d'être supprimé de 0.5.

4.4 Recherche d'optima locaux

Dans une troisième phase, une recherche d'optimum local est lancée. Autrement dit, un calcul du voisinage est effectué afin de trouver le motif maximisant la mesure de qualité dans le voisinage. Une première méthode possible est de calculer l'ensemble des voisins directs verticaux et horizontaux. Le voisin direct maximisant la mesure de qualité cible est choisi. Cette procédure est répétée jusqu'à ne plus avoir de voisins qui aient une meilleure mesure de qualité : on a trouvé un optimum local.

On pourrait également ne calculer que les voisins directs verticaux. Ceci permettrait de limiter le nombre d'éléments voisins (et donc leur $WRAcc$) à calculer. De plus tout élément appartenant au voisinage direct horizontal est atteignable avec la sélection successive de deux éléments du voisinage direct vertical (suppression puis addition d'item ou l'inverse). Enfin, étant donné le grand facteur de branchement dans l'espace de recherche, le calcul de l'ensemble des voisins et de leurs mesures (souvent dépendante du calcul du support) est coûteux. Pour contrer cela, une technique est de générer aléatoirement un nombre de voisins compris entre 1 et le nombre réel de voisins directs, valant, pour m items possibles et n itemsets dans la séquence, $m(2n+1)$ (voir preuve ci-dessous).

Démonstration : Pour une séquence s, soit n sa longueur (son nombre d'itemsets), k sa taille, et m le nombre d'items possibles. On a alors :

$$
\begin{aligned}
|VoisinsDirects| &= |VoisinsSuppression| + |Sextension| + |Iextension| \\
&= k + m(n+1) + |Iextension| \\
|Iextension| &= \sum_{i=1}^{n} |\{itemsPossibles\}| - |\{itemset\}_i| \\
&= nm - \sum_{i=1}^{n} |\{itemset\}_i| = nm - k \\
|VoisinsDirects| &= m(2n+1)
\end{aligned}
$$

4.5 Filtrage

Afin de limiter la redondance des motifs trouvés, une procédure de filtrage est nécessaire, même si la composante aléatoire de notre algorithme permet déjà une bonne diversité des résultats. Une première possibilité, que nous utilisons, est de stocker l'ensemble des résultats obtenus (optima locaux) dans une structure de données, et de les filtrer en fin d'algorithme à la manière de Bosc et al. (2018).

Une autre possibilité, est de stocker les motifs dans une file de priorité F de taille k. A chaque fois qu'un nouveau motif est trouvé, si sa qualité est meilleure que le minimum des motifs de F, alors il est ajouté, et le ou les autres éléments similaires dans F sont supprimés. Cette technique permet de minimiser l'utilisation de mémoire utilisée. Notre coût mémoire est alors minimal. On constate cependant que les résultats sont de moins bonne qualité via ce filtrage heuristique.

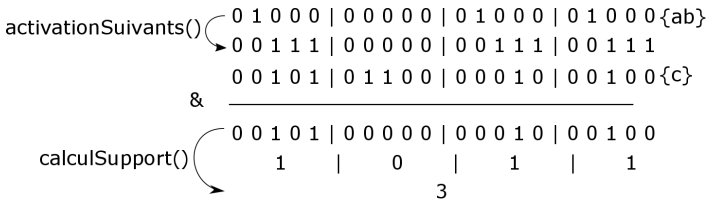

FIG. 2: Visualisation de CalculSupport pour la séquence $\langle\{ab\},\{c\}\rangle$.

4.6 Représentation verticale

Des algorithmes de découverte de motifs séquentiels comme SPAM (Ayres et al. (2002)) utilisent une représentation verticale afin d'améliorer le temps de calcul. Cette technique suppose un ordre lexicographique sur l'énumération, permettant de réutiliser une représentation verticale d'un motif pour l'étendre (I ou S-extension) en ajoutant un élément *en fin de séquence*. Dans notre cas, on doit pouvoir ajouter un élément à tout endroit de la séquence. Il faut donc pouvoir calculer la représentation verticale sans cette pré-condition. Pour cela nous proposons de garder dans une table de hashage l'ensemble des représentations verticales des itemsets rencontrés (mémoïsation) et nous les recombinons unes par unes afin de construire la représentation verticale de la séquence voulue. Dans le cas où un ensemble d'éléments nouveau est rencontré, nous le calculons.

Un exemple est donné dans Figure 2. Dans ce cas, le motif $\langle\{ab\},\{c\}\rangle$ a été généré. Imaginons que $\langle\{c\}\rangle$ soit un motif déjà généré, mais pas $\langle\{ab\}\rangle$. Nous sommes donc dans un cas qui n'est pas géré par SPAM, puisque l'on ajoute un nouvel itemset avant une séquence dont on connaît la représentation verticale. L'algorithme de calcul du support va d'abord chercher à trouver la représentation verticale de {ab}. Puisqu'elle n'a pas encore été trouvée, elle va être générée et ajoutée à la structure de mémoïsation. Une fois ceci fait, les représentations verticales sont combinées comme décrit dans Figure 2.

5 Expérimentations

Nous évaluons empiriquement notre approche, à travers les questions suivantes : Quelle est la qualité de nos résultats comparativement à l'état de l'art ? La représentation verticale permet-elle d'obtenir un gain de performances ? Comment évolue la consommation mémoire ? Est-il plus intéressant d'effectuer une recherche d'optimum local en calculant seulement le voisinage direct vertical ou les voisinages directs verticaux et horizontaux ?

Pour l'évaluation empirique, nous évaluons notre algorithme baptisé `HillSeqS` sur divers jeux de données (décrits dans le tableau 3) : **promoters**, **splice** sont issus du célèbre répertoire UCI [1], **aslbu**, **block** et **context** sont associés à la publication (Mörchen et Ultsch (2007)). Enfin, nous utilisons **sc2** un jeu de données associé à la publication (Bosc et al. (2017)). Dans ce dernier, chaque itemset correspond à des actions d'un joueur sur Starcraft 2 durant une fenêtre de temps avec une étiquette qui correspond au type de la faction gagnante.

1. http ://archive.ics.uci.edu/ml/index.php

Dataset	$\|Seq\|$	$\|items\|$	kmax
aslbu	441	126	30
promoters	106	4	58
splice	3190	8	62
blocks	210	8	13
context	240	47	124
sc2	500	25	31

FIG. 3: Jeux de données.

Dataset	$Naive$	$Bitset$	amélioration(%)
aslbu	141	709	402
promoters	161	472	193
splice	4	12	200
blocks	7896	48612	515
context	16	22	37
sc2	126	1695	1245

FIG. 4: Nombre d'itérations pour 180 s.

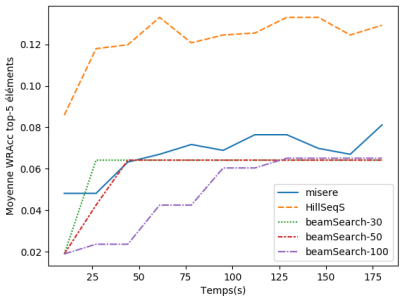

FIG. 5: WRAcc en fonction du temps.

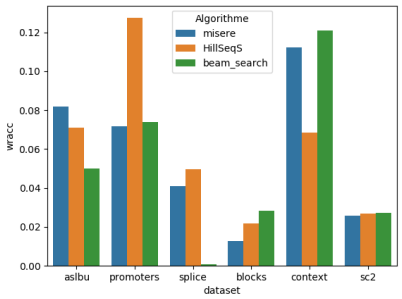

FIG. 6: WRAcc moyenne pour k=5, t=180s.

Notre approche est au pire comparable aux autres algorithmes, au mieux supérieure, et ce sans nécessiter de paramétrage. Elle s'avère relativement stable selon le jeu de données, contrairement aux expérimentations avec notre implémentation `Beam Search` paramétrée pour un nombre de faisceaux de 30, une valeur qui donne de bons résultats en moyenne. Notons qu'un "Beam Search" souffre du problème de redondance des motifs et que, pour des comparaisons plus justes, nous avons ajouté l'étape de filtrage à chaque étape de la recherche en faisceau. Notre algorithme est moins performant sur le jeu **context** qui possède des séquences très longues. Une explication pourrait être que ce jeu de données possède des optima locaux "longs" à atteindre, et que beaucoup d'entre eux sont de mauvaises qualités. Une amélioration possible serait donc de relancer la recherche en cas "d'enfermement" dans la recherche d'optimum local.

La recherche d'optimum local peut être trop longue de sorte que le nombre d'optima locaux trouvés peut être inférieur à k. Si le peu de motifs trouvés sont de mauvaises qualités, ce qui est le cas pour le jeu **context**, la moyenne des WRAcc sera plus faible. Une piste d'amélioration pourra être de sortir de la recherche d'optimum si l'amélioration de la qualité du motif courant n'est pas significative. Le graphique Figure 5 présente les moyennes des WRAcc pour les 5 meilleurs motifs non-redondants trouvés et différents budgets de temps, sur le jeu **promoters**. A chaque pas de temps, chaque algorithme a été relancé et c'est pourquoi on observe parfois une légère diminution locale de la qualité des résultats sur les approches à composante aléatoire `misère` et `HillSeqS`. On constate une augmentation avec le temps de la qualité des résultats pour notre approche. Ceci est dû au fait que chaque tirage garantit la découverte d'un optimum local : on exploite des zones de l'espace de recherche de manière aléatoire et un budget temps plus important permet une plus grande exploration. Notons cependant que notre algorithme, tout comme `misère`, ne garantit pas une recherche exhaustive.

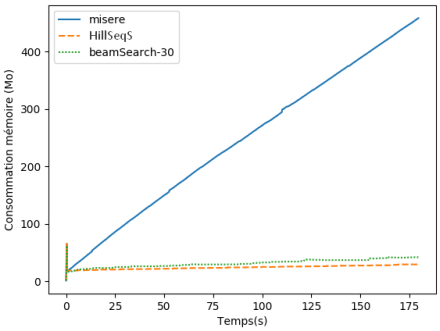

FIG. 7: Consommation mémoire.

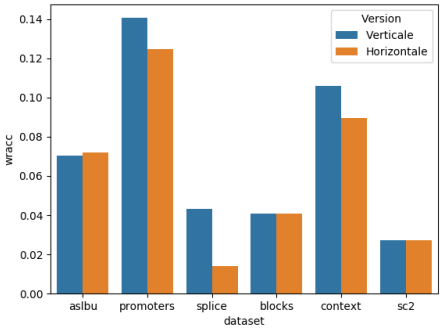

FIG. 8: Évaluation des voisinages.

Les consommations en mémoire en fonction du temps pour `HillSeqS`, pour `misère` et pour `Beam Search` sont présentés dans Figure 7. Le jeu de données utilisé est **sc2**. La consommation mémoire de notre algorithme est très faible : nous ne stockons dans la file de priorité que les optima locaux, un nombre négligeable au vu de tous les motifs calculés. La consommation mémoire de `misère` augmente au fil des tirages effectués. Ceci s'explique par notre implémentation qui reprend le pseudo-code de (Egho et al. (2017)) où chaque élément est ajouté dans une file de priorité : tous les éléments sont conservés. Le pic observé sur Figure 7 est dû à la mise en mémoire des jeux de données.

Le graphique Figure 8 montre que le calcul du voisinage direct vertical uniquement (en bleu) donne de meilleurs résultats en moyenne que le calcul du voisinage direct vertical et horizontal. Une explication pourrait être que le changement d'un item de la séquence (parcours horizontal) est atteignable en deux étapes verticales : la suppression et l'ajout d'un item. Ainsi, on diminue le nombre de calculs de voisins lors de la recherche d'un optimum local, ce qui augmente le nombre d'échantillonnages possibles, menant à des résultats plus intéressants en moyenne. De plus, on peut voir dans Table 4 que dépendamment du jeu de données, la représentation verticale permet un net gain de performances : la représentation sous forme de bitsets permet d'augmenter le nombre d'itérations, mais cette augmentation est variable. Ceci peut s'expliquer par la composante aléatoire de l'algorithme, et par la « forme » des zones autour des optima locaux : la vitesse de convergence vers un optimum local est variable. Un autre paramètre influençant cette hétérogénéité des augmentations est le nombre d'itemsets possibles. A chaque nouvel itemset trouvé, un calcul sur tout le jeu de données est effectué, ce qui est beaucoup plus coûteux que de faire appel à la structure de mémoization. De même, la longueur des séquences influence l'efficacité de la représentation sous forme de bitsets, et ce potentiellement différemment de la façon dont elle influence l'implémentation naïve, ceci pouvant expliquer la variation des augmentations.

Le motif suivant est donné par notre algorithme sur le jeu de données sc2. Il correspondant à une suite de constructions lors d'une partie et caractérise la victoire du joueur de type "Terran" lors des matchs "Terran vs Zerg", avec une WRAcc de 0.048 :

$$\langle \{Barracks\}, \{SupplyDepot\}, \{Hatchery\}, \{SpineCrawlerBarrack\}, \{Barracks\} \rangle$$

Starcraft 2 est un jeu de stratégie où il existe un dilemme entre la production d'unités de combat et l'économie à plus long terme. Ici, on constate que le joueur Terran construit des bâtiments de production d'unité de combat ("Barracks"), et que le joueur Zerg préfère construire un bâtiment permettant d'améliorer son économie sur le long terme ("Hatchery"), au détriment de ses capacités militaires sur le court terme. La construction d'un "Spine Crawler", un bâtiment militaire défensif Zerg, ne permet pas de compenser suffisamment l'avantage militaire du joueur Terran, qui agresse son adversaire en début de partie, et finit par gagner. Cet exemple nous montre que notre algorithme produit des motifs qui peuvent être pertinents. Ici nous l'avons utilisé afin de mieux expliquer le système "jouer une partie", mais on pourrait aussi imaginer, entre autres, l'utiliser dans un outil prédictif permettant, au cours d'une partie, de prédire son issue, utile pour les paris en ligne, ou bien pour améliorer l'entraînement des joueurs, par exemple.

6 Conclusion

Nous avons décrit un travail pour la découverte de sous-groupes depuis des séquences d'itemsets étiquetées. Nous développons des propositions récentes adaptées aux séquences d'items avec `Misère` (Egho et al. (2017)) pour un échantillonnage de motifs suivi d'un processus d'exploration locale et une extraction d'optima locaux. Notre méthode donne des résultats à tout moment de l'exécution, permet la découverte d'optima locaux sans paramétrage et bénéficie de la recherche aléatoire pour limiter la redondance dans les résultats. Différents axes d'amélioration feront l'objet de futurs travaux. L'impact de nouvelles stratégies de tirage doit être testé. La généralisation des séquences tirées pourrait exploiter des connaissances statistiques extraites par un pré-traitement (par exemple, favoriser la suppression d'un item peu présent dans les séquences étiquetées par la classe cible). De plus, notre recherche d'optima locaux doit pouvoir être améliorée, notamment pour empêcher l'algorithme de s'enfermer dans une zone de l'espace où la recherche des sous-groupes de qualité s'avère trop lente.

Références

Atzmüller, M. et F. Puppe (2006). Sd-map – a fast algorithm for exhaustive subgroup discovery. In *Proceedings PKDD 2006*, pp. 6–17.

Ayres, J., J. Flannick, J. Gehrke, et T. Yiu (2002). Sequential pattern mining using a bitmap representation. In *Proceedings ACM SIGKDD 2002*, pp. 429–435.

Boley, M., C. Lucchese, D. Paurat, et T. Gärtner (2011). Direct local pattern sampling by efficient two-step random procedures. In *Proceedings ACM SIGKDD 2011*, pp. 582–590.

Bosc, G., J.-F. Boulicaut, C. Raïssi, et M. Kaytoue (2018). Anytime discovery of a diverse set of patterns with monte carlo tree search. *DMKD 32*(3), 604–650.

Bosc, G., P. Tan, J.-F. Boulicaut, C. Raïssi, et M. Kaytoue (2017). A Pattern Mining Approach to Study Strategy Balance in RTS Games. *IEEE Transactions on Computational Intelligence and AI in games 9*(2), 123–132.

Dafé, G., A. Veloso, M. Zaki, et W. Meira (2015). Learning sequential classifiers from long and noisy discrete-event sequences efficiently. *DMKD 29*(6), 1685–1708.

Diop, L., C. T. Diop, A. Giacometti, D. H. Li, et A. Soulet (2018). Echantillonnage de motifs séquentiels sous contrainte sur la norme. In *Proceedings EGC 2018*, pp. 35–46.

Duivesteijn, W., A. J. Feelders, et A. Knobbe (2016). Exceptional model mining. *Data Mining and Knowledge Discovery 30*(1), 47–98.

Egho, E., D. Gay, M. Boullé, N. Voisine, et F. Clérot (2017). A user parameter-free approach for mining robust sequential classification rules. *KAIS 52*(1), 53–81.

Lavrac, N., P. A. Flach, et B. Zupan (1999). Rule evaluation measures : A unifying view. In *Proceedings ILP 1999*, pp. 174–185.

Mörchen, F. et A. Ultsch (2007). Efficient mining of understandable patterns from multivariate interval time series. *Data Mining and Knowledge Discovery 15*(2), 181–215.

Novak, P. K., N. Lavrač, et G. I. Webb (2009). Supervised descriptive rule discovery : A unifying survey of contrast set, emerging pattern and subgroup mining. *Journal Machine Learning Research 10*, 377–403.

Toivonen, H. (1996). Sampling large databases for association rules. In *Proceedings VLDB 1996*, pp. 134–145.

Wrobel, S. (1997). An algorithm for multi-relational discovery of subgroups. In *Proceedings PKDD 1997*, pp. 78–87.

Zaki, M. J. (2001). Spade : An efficient algorithm for mining frequent sequences. *Machine Learning 42*(1), 31–60.

Zhou, C., B. Cule, et B. Goethals (2016). Pattern based sequence classification. *IEEE Transactions on Knowledge and Data Engineering 28*, 1285–1298.

Summary

Discovering rules that characterize classes remains difficult, especially within the context of sequential data analysis. This has been nicely formalized within the subgroup discovery setting and numerous algorithms have been proposed over the last 20 years. An exhaustive enumeration strategy is generally intractable. Therefore, heuristic approaches are needed and the reference framework relies on a beam search stategy and its run time parameters. We propose a sampling method that samples patterns from the search space to support subgroup discovery in labeled sequences of itemsets. Our approach enables the discovery of local optima with respect to a quality measure though the method remains generic with respect to the chosen quality measure. We do not have to set parameters and it is simple to implement. Our empirical validation that includes a comparison with state-of-the-art algorithms exhibits interesting results.