# Echantillonnage d'itemsets à forte utilité moyenne sous contraintes de taille

Lamine Diop

Université de Tours, France, lamine.diop@univ-tours.fr

**Résumé.** Les algorithmes d'extraction d'High-Utility Itemset (HUI) sont des méthodes permettant de découvrir des connaissances dans une base de données où les items sont pondérés. Leur importance a été largement démontrée dans de nombreuses applications réelles. Dans cet article, nous proposons un algorithme nommé HAISAMPLER qui échantillonne des itemsets à forte utilitè moyenne sous contraintes de taille afin d'éviter les itemsets longs et rares, formés d'items de faible poids. Les expérimentations montrent qu'il tire en quelques secondes des milliers de motifs à forte utilité moyenne sur différents jeux de données.

## 1 Introduction

La découverte d'High-Utility Itemset (HUI) (Yao et al., 2004) est une extension du problème de la découverte de motifs fréquents (Agrawal et Srikant, 1994) qui prend en compte à la fois la quantité des items dans une transaction et leurs qualités (le prix par exemple). Son importance a été largement prouvée dans de nombreuses applications réelles telles que l'analyse du comportement des utilisateurs (Shie et al., 2013), la découverte interactive de motifs (Ahmed et al., 2009), etc. La découverte interactive est un processus qui nécessite une interaction rapide entre le système et l'utilisateur (van Leeuwen, 2014). Malheureusement, l'efficacité de ces méthodes d'extraction exhaustive d'HUI dépend souvent de la taille des bases de données sur lesquelles elles sont appliquées. Des méthodes dites top-k (Singh et al., 2019) sont également utilisées pour trouver les meilleurs motifs. Par conséquent, elles se concentrent souvent sur la même partie qui contient des motifs légèrement différents et conduit ensuite à un manque de diversité. En effet, cette dernière est cruciale pour présenter à l'utilisateur un ensemble de motifs variés à chaque itération afin d'améliorer sa vision des données et aider le système à connaître son intérêt. Un autre problème que rencontrent les méthodes exactes de fouille de motifs à forte utilité est celui de la longue traîne où des motifs contenant des items de faible poids ont des utilités élevées du fait de leur longueur. Dans (Lin et al., 2016), les auteurs proposent une mesure d'utilité basée sur la moyenne pour éviter le problème de la longue traîne. Cependant, celle-ci favorise les itemsets ayant une longueur égale à 1, car ils ne sont pas affectés par la division, et donc les motifs retournés deviennent évidents pour l'utilisateur. Une alternative consiste à poser un seuil de fréquence minimale afin de contraindre les motifs longs, mais il est très difficile pour l'utilisateur de fixer le seuil d'intérêt minimal tel que la fréquence.

Pour résoudre ce problème, nous proposons de bénéficier des techniques d'échantillonnage en sortie sous contraintes de taille (Diop, 2020). L'échantillonnage en sortie consiste à fournir un échantillon représentatif de motifs selon une distribution proportionnelle à une mesure

d'intérêt choisie par l'utilisateur tout en assurant une bonne diversité entre les motifs retournés. Boley et al. (2012) ont étendu leurs travaux antérieurs pour prendre en compte les poids des items. Mais leur méthode tire chaque motif avec une probabilité proportionnelle à la fréquence multipliée par le poids du motif qui est calculé indépendamment de toute transaction. En outre, elle ne marche pas avec des contraintes de taille. De même, Diop et al. (2022) pondèrent chaque motif avec une utilité fondée sur la taille (indépendamment de toute transaction) en intégrant des contraintes de taille pour éviter le phénomène de la longue traîne. Dans cet article, nous proposons une méthode originale d'échantillonnage en sortie d'itemsets à forte utilité moyenne intégrant des contraintes de taille. Contrairement aux méthodes qui se basent sur des algorithmes heuristiques (Song et al., 2021) pour trouver les top-k itemsets à forte utilité, la méthode d'échantillonnage que nous proposons tire chaque itemset ayant respecté les contraintes de tailles avec une probabilité exactement égale à son utilité divisée par la somme des utilités moyennes de tous les itemsets de la base de données qui respectent les contraintes de taille. A notre connaissance, cet article propose la première méthode destinée à l'échantillonnage d'itemsets à forte utilité moyenne en plus d'intégrer des contraintes de taille.

## 2 Préliminaires et formulation du problème

Soit $\mathcal{I} = \{e_1, \cdots, e_N\}$ un ensemble fini de littéraux nommés items munis d'une relation d'ordre total $>_\mathcal{I} : e_1 >_\mathcal{I} \cdots >_\mathcal{I} e_N$. Un itemset ou motif $X = e_{i_1} \cdots e_{i_n}$, avec $n \leq N$, est un sous ensemble non vide de $\mathcal{I}$, $X \subseteq \mathcal{I}$. Le langage de motifs correspond à $\mathcal{L} = 2^{|\mathcal{I}|} \setminus \emptyset$ et la taille d'un motif $X \in \mathcal{L}$ notée par $|X|$ est le nombre d'items qu'il contient. Une base de données transactionnelles $\mathcal{D}$ correspond à un ensemble de couple $(j, t)$ où $j \in \mathbb{N}$ est l'unique identifiant d'une transaction et $t = e_1 \cdots e_n$ est un itemset dans $\mathcal{I}$ de taille $|t| = n$. Nous notons par $\mathcal{L}(\mathcal{D})$ l'ensemble des motifs qui apparaissent dans $\mathcal{D}$. Dans la suite de cet article, une transaction d'identifiant $j$ est notée par $t_j$. En outre, pour une transaction $t_j = e_{j_1} \cdots e_{j_n}$, nous notons $t_j^i = e_{j_{i+1}} \cdots e_{j_n}$ un itemset formé en éliminant les $i$ premiers items de $t$. On a donc $|t_j^i| = |t_j| - i$. Le i-ème item de la transaction $t$ est $t_j[i] = e_{j_i}$. Dans cet article, chaque item $e_{j_i}$ d'une transaction $t_j$ a un poids, un nombre réel strictement positif, qui dépend de cette transaction, appelé son utilité. Par exemple, dans le cas d'une transaction qui représente l'ensemble des éléments achetés par un client, l'utilité d'un item $e_i$ dans la transaction $t$ peut être le produit de sa quantité $q(e_i, t)$ et de son prix unitaire $p(e_i)$. Pour être plus simple sur le reste de cet article, nous associons chaque item $e_i$ avec sa quantité dans la transaction $t$ qui le contient, $e_i : q(e_i, t)$. Le tableau 1 montre une base de données $\mathcal{D}$ composée de 5 transactions $t_1$, $t_2$, $t_3$, $t_4$ et $t_5$ définies sur l'ensemble des items $\mathcal{I} = \{A, B, C, D, E, F\}$. Nous supposons que $A >_\mathcal{I} B >_\mathcal{I} C >_\mathcal{I} D >_\mathcal{I} E >_\mathcal{I} F$. Dans la base de données $\mathcal{D}$, les prix sont : $p(A) = 25$, $p(B) = 30$, $p(C) = 10$, $p(D) = 5$, $p(E) = 15$ et $p(F) = 10$. Avec $t_1$, on a les quantités $q(A, t_1) = 2$, $q(B, t_1) = 3$ et $q(C, t_1) = 2$.
Cette base de données sera utilisée par la suite pour donner des illustrations dans le reste de cet article. Etant donné que les items de la transaction $t_1$ n'ont pas le même poids, alors les occurrences de motifs de $t_1$ peuvent ne pas avoir la même utilité dans $t_1$.

**Définition 1 (Occurrence de motif)** *Soit $X$ un motif du langage $\mathcal{L}$ d'une base de données $\mathcal{D}$. S'il existe une transaction $t_j$ de $\mathcal{D}$ telle que $X \subseteq t_j$, alors $X_j$ est une occurrence du motif $X$*

| $\mathcal{D}$ | | | | | | |
|---|---|---|---|---|---|---|
| $t_1$ | A :2 | B :3 | C :2 | | | |
| $t_2$ | | B :2 | | D :4 | | |
| $t_3$ | A :1 | | C :1 | D :1 | | |
| $t_4$ | A :3 | B :1 | | | | |
| $t_5$ | A :1 | B :2 | | D :1 | E :1 | F :1 |

| Items | Prix |
|---|---|
| A | 25 |
| B | 30 |
| C | 10 |
| D | 5 |
| E | 15 |
| F | 10 |

TAB. 1 – *Exemple de base de données $\mathcal{D}$ avec des utilités sur les items*

*dans la transaction $t_j$. L'utilité du motif $X$ dans la transaction $t_j$, notée par $\texttt{uOcc}(X, t_j)$, est égale à 0 si $X \not\subseteq t_j$ ou $X = \emptyset$, sinon $\texttt{uOcc}(X, t_j) = \sum_{e \in X} (q(e, t_j) \times p(e))$.*

Il existe également des utilités qui sont indépendantes de toute base de données telles que celles dites fondées sur la taille (Diop, 2020). Dans la suite, nous considérons l'utilité fondée sur la taille définie par $\texttt{uLen}_{[m..M]}(X) = 1/|X|$ si $|X| \in [m..M]$ et 0 sinon, avec $m$ et $M$ des entiers positifs tels que $m \leq M$. Ainsi, un motif dont la taille est plus grande que $M$ ou plus petite que $m$ sera jugé inutile.

**Définition 2 (Utilité moyenne d'un motif sous contraintes de taille)** *Soit $\mathcal{D}$ une base de données, $\mathcal{L}$ son langage, $m$ et $M$ des entiers tels $m \leq M$. L'utilité moyenne du motif $X \in \mathcal{L}$ dans la base de données $\mathcal{D}$ sous contraintes de taille minimale $m$ et maximale $M$, notée par $u^{moy}_{[m..M]}$, est égale au produit de la somme des utilités de ses occurrences et de son utilité fondée sur la taille. Formellement, on a : $u^{moy}_{[m..M]}(X, \mathcal{D}) = (\sum_{(j,t) \in \mathcal{D} \wedge X \subseteq t} \texttt{uOcc}(X, t)) \times \texttt{uLen}_{[m..M]}(X)$.*

Etant donné $\mathcal{D}$ une base de données transactionnelles avec des poids sur les items (quantité et/ou qualité), des contraintes de taille minimale $m$ et maximale $M$, notre objectif est de tirer un motif $X$ du langage $\mathcal{L}$ avec une probabilité exactement égale à : $\mathbb{P}(X, \mathcal{D}) = u^{moy}_{[m..M]}(X, \mathcal{D}) / \sum_{X' \in \mathcal{L}(\mathcal{D})} u^{moy}_{[m..M]}(X', \mathcal{D})$.

# 3 Echantillonnage d'itemsets à forte utilité en deux phases

Dans cette section, nous allons présenter d'abord les bases de notre méthode avant de présenter l'algorithme HAISAMPLER qui échantillonne des itemsets à forte utilité moyenne.

## 3.1 Bases de la méthode

La méthode d'échantillonnage d'itemsets à forte utilité que nous proposons dans cet article utilise un système de pondération par taille basé sur la position afin de pondérer chaque transaction. C'est un système qui consiste à pondérer chaque item d'une transaction donnée suivant la position qu'il y occupe et suivant les tailles des motifs auxquels il apparaît comme premier item selon la relation d'ordre total $>_{\mathcal{I}}$. Les poids des items sont ensuite exploités pour tirer un motif en utilisant une probabilité conditionnelle. [1]

**Pondération d'une transaction :** Soient $t$ une transaction de taille $n$ définie sur un ensemble d'items $\mathcal{I}$ munis d'une relation d'ordre totale $>_{\mathcal{I}}$, $m$ et $M$ des contraintes de taille minimale

1. Preuves : `https://github.com/HAISampler/haisampler/blob/main/preuves.pdf`

et maximale respectivement. Le i-ème item de la transaction $t$, $t[i]$, est associé à deux listes de valeurs $\omega_\ell^+(t[i],t)$ et $\omega_\ell^-(t[i],t)$, pour $\ell \in [m..M]$. Par convention $\binom{k}{n}$ =0 si k>n et 1 si k=0.

**Définition 3** *Le poids $\omega_\ell^+(t[i],t)$ est la somme des utilités des occurrences de motifs de taille $\ell-1$ dans la transaction $t^i = t[i+1]\cdots t[n]$ auxquelles on adjoint l'item $t[i]$ et le poids $\omega_\ell^-(t[i],t)$ est celle des occurrences de motifs de taille $\ell$ dans $t^i$. Formellement, on a :* $\omega_\ell^+(t[i],t) = \sum_{X\subseteq t^i \wedge |X|=\ell-1} u\mathit{Occ}(\{t[i]\}\cup X,t)$ *et* $\omega_\ell^+(t[i],t) = \sum_{X\subseteq t^i\wedge|X|=\ell} u\mathit{Occ}(X,t)$

La propriété 1 donne une formalisation des poids à l'aide de la définition 3.

**Propriété 1 (Calcul des $\omega_\ell^\bullet(t[i],t)$)** *Les poids $\omega_\ell^+(t[i],t)$ et $\omega_\ell^-(t[i],t)$ de l'item $t[i]$, pour tout $\ell \in [m..M]$, peuvent être formellement écrits comme suit :* $\omega_\ell^+(t[i],t) = \omega_1(t[i],t)\times\binom{\ell-1}{|t^i|} + \sum_{\star\in\{+,-\}}\omega_{\ell-1}^\star(t[i+1],t)$ *et* $\omega_\ell^-(t[i],t) = \sum_{\star\in\{+,-\}}\omega_\ell^\star(t[i+1],t)$ *avec* $\omega_1^+(t[i],t) = u\mathit{Occ}(t[i],t)$ *pour tout* $i\in[1..|t|]$ *et* $\omega_\ell^\star(t[i],t)=0$ *pour tout* $i>|t|$.

En effet, les poids d'un items $t[i]$ se déduisent de ceux de $t$ [i+1]. A l'aide de la propriété 1, nous pouvons facilement calculer le poids d'une transaction suivant des contraintes de taille. Par définition, l'utilité moyenne d'une occurrence de motif $X\subseteq t$ est égale à $\mathtt{uOcc}(X,t)/|X|$.

**Propriété 2 (Pondération d'une transaction)** *Le poids de la transaction $t$ sous contraintes de taille minimale $m$ et maximale $M$, noté $\omega_{[m..M]}^{umoy}(t)$, est la somme des utilités moyennes des occurrences de motifs qu'elle contient. Formellement, on a :*

$$\omega_{[m..M]}^{umoy}(t) = \sum_{\ell=m}^{M}\left(\frac{1}{\ell}\sum_{i=1}^{|t|}\omega_\ell^+(t[i],t)\right) = \sum_{\ell=m}^{M}\frac{1}{\ell}\left(\omega_\ell^+(t[1],t)+\omega_\ell^-(t[1],t)\right).$$

**Exemple 1** *On a :* $t_1$ : {

| | *A* | | | *B* | | | *C* | | |
|---|---|---|---|---|---|---|---|---|---|
| $\omega_\ell^+$ | *50* | *210* | *160* | *90* | *110* | *0* | *20* | *0* | *0* |
| $\omega_\ell^-$ | *110* | *110* | *0* | *20* | *0* | *0* | *0* | *0* | *0* |

}

*Donc* $\omega_{[1..3]}^{umoy}(t_1) = (50+110)/1 + (210+110)/2 + (160+0)/3 = 373.33$.

**Tirage d'un itemset à partir d'une transaction :** Le tirage d'un motif à partir d'une transaction pondérée à base de position peut se faire en utilisant la probabilité conditionnelle. Le lemme 1 donne une idée sur le calcul de la probabilité de tirer un item donné sachant qu'on a déjà tiré (ou pas) des items supérieurs selon la relation d'ordre $>_\mathcal{I}$ introduite à la section 2.

**Lemme 1** *Soient $\ell$ la taille de l'itemset à tirer, $\mathbb{P}_\ell^t(t[i]|X,\ell')$ la probabilité de tirer l'item $t[i]$ de la transaction $t$ après y avoir tiré $\ell-\ell'$ items et les avoir stockés dans $X$, avec $e >_\mathcal{I} t[i]$ pour tout $e\in X$. La probabilité de tirer $t[i]$ sachant $X$ et $\ell'$ peut être formulée comme suit :*

$$\mathbb{P}_\ell^t(t[i]|X,\ell') = \frac{\sum_{X'\subseteq t^i\wedge|X'|=\ell'-1} u\mathit{Occ}(X\cup\{t[i]\}\cup X',t)}{\sum_{X'\subseteq t^{i-1}\wedge|X'|=\ell'} u\mathit{Occ}(X\cup X',t)}.$$

**Propriété 3** *La probabilité de tirer l'item $t[i]$ de la transaction $t$ sachant l'itemset $X$ et la taille $\ell'$, avec $|X|=\ell-\ell'$, notée par $\mathbb{P}_\ell^t(t[i]|X,\ell')$, est donnée par la formule suivante :*

$$\mathbb{P}_\ell^t(t[i]|X,\ell') = \frac{\left(\sum_{k<i\wedge t[k]\in X}\omega_1(t[k],t)\right)\times\binom{\ell'-1}{|t^i|} + \omega_{\ell'}^+(t[i],t)}{\left(\sum_{k<i\wedge t[k]\in X}\omega_1(t[k],t)\right)\times\binom{\ell'}{|t^{i-1}|} + \left(\sum_{\star\in\{+,-\}}\omega_{\ell'}^\star(t[i],t)\right)}.$$

*La probabilité que l'item $t[i]$ ne soit pas tiré sachant $X$ et $\ell'$ est égale* $1-\mathbb{P}_\ell^t(t[i]|X,\ell')$.

Les preuves de ces deux formules découlent du fait que la probabilité de tirer $t[i]$ dépend des utilités des items déjà tirés et celles des items qui le suivent pour former un motif de taille $\ell$.

**Exemple 2** *Supposons que nous devons tirer un motif de taille $\ell = 2$ à partir de la transaction $t_1$. La probabilité de tirer $X = AC$ est calculée comme suit :* $\mathbb{P}^{t_1}(AC|\ell) = \mathbb{P}^{t_1}_{\ell}(A|X = \emptyset, \ell' = 2) \times (1 - \mathbb{P}^{t_1}_{\ell}(B|X = A, \ell' = 1)) \times \mathbb{P}^{t_1}_{\ell}(C|X = A, \ell' = 1)$. *Ce qui nous donne* $\mathbb{P}^{t_1}(AC|\ell) = \frac{0+210}{0+320} \times (1 - \frac{50\times\binom{1-1}{1}+90}{50\times\binom{1}{2}+90+20}) \times \frac{50\times\binom{1-1}{0}+20}{50\times\binom{1}{1}+20} = \frac{210}{320} \times (1 - \frac{140}{210}) \times \frac{70}{70} = \frac{210}{320} \times \frac{70}{210} \times \frac{70}{70} = \frac{70}{320}$.

## 3.2 HAISAMPLER : échantillonnage d'itemsets à forte utilité moyenne

**Algorithm 1** HAISAMPLER (High Average-utility Itemset Sampler)

**Input :** Une base de données transactionnelles $\mathcal{D}$ ayant des poids sur les items munis d'une relation d'ordre total $>_{\mathcal{I}}$ et des contraintes de taille minimale $m$ et maximale $M$
**Output :** Un motif $X$ tiré proportionnellement à son utilité moyenne : $X \sim u^{moy}_{[m..M]}(\mathcal{L}, \mathcal{D})$
//Phase de pondération des transactions de la base de données
1: Calculer le poids de chaque transaction $t$ dans $\mathcal{D}$ : $\omega^{umoy}_{[m..M]}(t)$
//Phase de tirage d'un itemset
2: Tirer une transaction proportionnellement à son poids : $t \sim \omega^{umoy}_{[m..M]}(\mathcal{D})$
3: Tirer une taille $\ell$ suivant son poids $\sum_{\star\in\{+,-\}} \omega^{\star}_{[\ell..\ell]}(t[1], t) : \ell \sim \omega^{umoy}_{[m..M]}(t)$
4: $X \leftarrow \emptyset$, $y \leftarrow 0$, $i \leftarrow 1$
5: **while** $\ell > 0$ **do**
6: $\quad z \leftarrow y \times \binom{\ell}{|t^i|} + \omega^{+}_{\ell}(t[i], t) + \omega^{-}_{\ell}(t[i], t)$
7: $\quad x \leftarrow random() \times z$ ▷ Tirer aléatoirement un nombre réel entre 0 et $z$
8: $\quad$ **if** $x \leq y \times \binom{\ell-1}{|t^i|} + \omega^{+}_{\ell}(t[i], t)$ **then**
9: $\quad\quad X \leftarrow X \cup \{t[i]\}$
10: $\quad\quad y \leftarrow y + \omega^{+}_{1}(t[i], t)$
11: $\quad\quad \ell \leftarrow \ell - 1$
12: $\quad i \leftarrow i + 1$
13: **return** $X$ ▷ Un motif tiré proportionnellement à son utilité moyenne dans $\mathcal{D}$

L'algorithme 1 prend en entrée une base de données transactionnelles définies sur un ensemble d'items munis d'une relation d'ordre totale $>_{\mathcal{I}}$ et des contraintes de tailles minimale $m$ et maximale $M$. On commence avec une phase de prétraitement qui calcule le poids de chaque transaction (ligne 1) en utilisant les propriétés 1 et 2. Pour le tirage d'un motif, on tire en premier lieu une transaction $t$ proportionnellement à son poids $\omega^{umoy}_{[m..M]}(t)$ (ligne 2). En deuxième lieu, on tire une taille $\ell$ proportionnellement à la somme des utilités moyennes des itemsets de taille $\ell$ que contient la transaction $t$ précédemment tirée (ligne 3). Les lignes 4 à 12 permettent de tirer aléatoirement un itemset de taille $\ell$ avec une probabilité proportionnellement à son utilité moyenne dans $t$. A la ligne 6, on calcule la somme totale, $z$, des utilités des itemsets qui commencent par $X \cup \{t[i]\}$ suivant la relation d'ordre $>_{\mathcal{I}}$ dans la transaction $t$. Ensuite, on tire aléatoirement un nombre réel entre 0 et $z$ (ligne 7). Si le nombre tiré est plus petit que la sommes des utilités des items qui, commençant par $X$, contiennent aussi l'item à la position $i$ de la transaction $t$, alors on ajoute $t[i]$ dans l'ensemble des items à retourner en sortie (lignes 8

et 9). Dans ce cas, la somme des utilités des items tirés est mise à jour dans la variable $y$ (ligne 10) et le nombre d'items à trouver diminue (ligne 11). Quand $\ell = 0$, on retourne un itemset $X$ tiré proportionnellement à son utilité moyenne dans la base de données $\mathcal{D}$ (ligne 13).

# 4 Expérimentations

Les expériences ont été conduites avec 3 jeux de données issus de l'UCI : `Adult`, `Chess` et `Mushroom` avec des versions pré-traitées (chaque item a été associé avec une utilité prise aléatoirement entre 1 et 100), et 3 bases de données réelles issus de SPMF `BMS`, `Foodmart` et `Retail`. Le tableau 2 détaille les caractéristiques des benchmarks. La valeur de la contrainte de taille minimale est fixée à $m = 1$ tout au long des expérimentations. Toutes les expériences ont été réalisées sur un PC de 2.11 GHz 2 Core CPU avec 32 Go de RAM. [2]

| $\mathcal{D}$ | $\lvert\mathcal{D}\rvert$ | $\lvert\mathcal{I}\rvert$ | $\lvert t\rvert_{\min}$ | $\lvert t\rvert_{\max}$ | $\lvert t\rvert_{moy}$ | $\omega(e,t)_{\min}$ | $\omega(e,t)_{\max}$ | $\omega(e,t)_{moy}$ |
|---|---|---|---|---|---|---|---|---|
| Adult | 48,842 | 97 | 12 | 15 | 14.87 | 1.0 | 99.0 | 50.04 |
| BMS | 59,602 | 497 | 1 | 267 | 2.51 | 7.0 | 9,000.0 | 724.79 |
| Chess | 28,056 | 58 | 7 | 7 | 7.00 | 1.0 | 99.0 | 50.05 |
| Foodmart | 4,141 | 1,559 | 1 | 14 | 4.42 | 50.0 | 2,166.0 | 655.66 |
| Mushroom | 8,124 | 90 | 22 | 23 | 22.69 | 1.0 | 99.0 | 50.02 |
| Retail | 88,162 | 16470 | 1 | 76 | 10.31 | 1.0 | 140.0 | 16.41 |

TAB. 2 – *Caractéristiques des jeux de données(nb transactions, nb items distincts, la taille minimale, maximale et moyenne des transactions, poids minimal, maximal et moyen des items)*

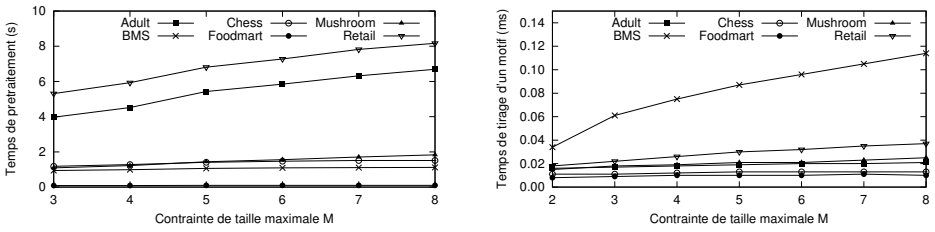
## 4.1 Rapidité de la méthode

Les temps moyens d'exécution que nous allons présenter ont été obtenus en répétant 100 fois le programme pour chaque cas. Nous avons omis les écart-types car ils sont très faibles.
*Temps de prétraitement :* La figure 1-(a) indique l'évolution du temps de prétraitement en fonction de la contrainte de taille maximale $M \in [3..8]$. On peut noter que le temps de prétraitement de HAISAMPLER augmente en fonction de la contrainte de taille maximale et de la taille de la base de données. Cependant, il reste inférieur à $9s$ pour tous nos jeux de données.
*Temps de tirage d'un motif :* La figure 1-(b) montre les courbes d'évolution du temps de tirage d'un motif en fonction de la contrainte de taille maximale $M \in [2..8]$ pour nos différents jeux de données. D'après ces courbes, on peut dire que le temps de tirage évolue légèrement en fonction de la contrainte de taille maximale et de la longueur maximale des transactions. Enfin, dans tous les cas, le temps de tirage d'un motif reste inférieur à $0.15\ ms$ sur tous les jeux de données. Il est même inférieur à $0.04\ ms$ quand $M \leq 8$ sauf dans `BMS`, ce qui veut dire que HAISAMPLER parvient à tirer des milliers de motifs en quelques secondes.

## 4.2 Impact des contraintes de taille sur les utilités moyennes des motifs

La figure 2 montre l'impact de la contrainte de taille maximale $M$ (5, 8 et $\infty$) en dressant la répartition des utilités moyennes des motifs en fonction de leur taille pour `BMS` et `Retail`.

2. HAISAMPLER est implémenté avec Python 3.8 `https://github.com/HAISampler/haisampler`.




(a) (b)

FIG. 1 – *Evolution du temps de prétraitement (a) et de tirage d'un motif (b) en fonction de $M$*

Les résultats montrent clairement que si la contrainte de taille maximale est très élevée voire absente, les motifs tirés sont très longs et formés que par des items à faible poids (c'est le phénomène de la longue traîne). D'ailleurs, aucun des motifs retournés par la méthode sans contrainte n'a une taille plus petite que 100 pour `BMS` et 20 pour `Retail`. On peut dire donc qu'il est très intéressant d'utiliser des contraintes de taille pour échantillonner des motifs à forte utilité à partir d'une base de données atteinte de la malédiction de longue traîne, ce qui est souvent le cas avec les données réelles.

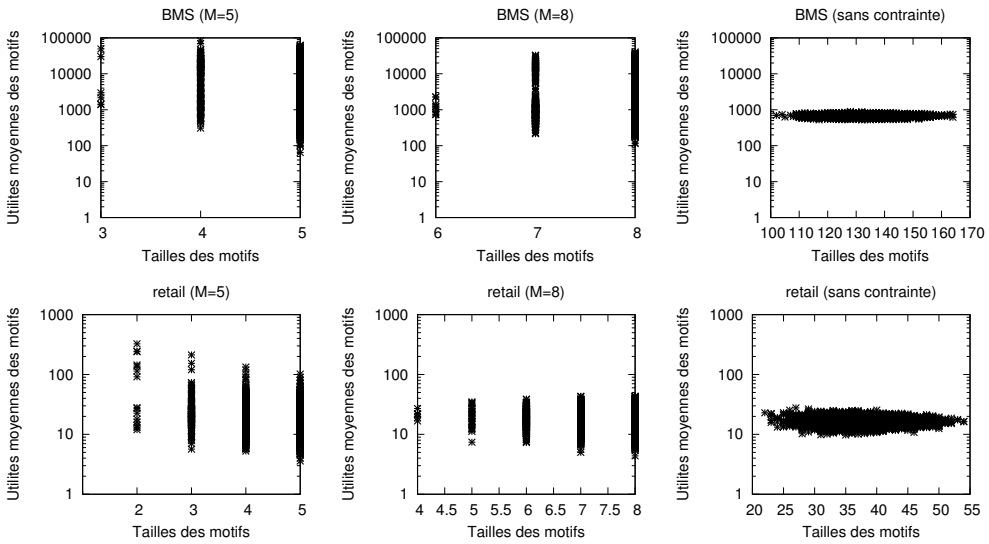



FIG. 2 – *Dispersion des utilités moyennes de 10,000 motifs échantillonnés*

# 5 Conclusion

Cet article présente la première méthode d'échantillonnage d'itemsets à forte utilité moyenne sous contraintes de taille. Nous avons montré que la méthode proposée est très efficace pour tirer des milliers de motifs en quelques secondes sur des jeux de données réels et synthétiques avec un temps de prétraitement raisonnable. Les expérimentations réalisées ont montré l'intérêt des contraintes de taille pour échantillonner des motifs ayant de fortes utilités moyennes.

Par ailleurs, on peut facilement adapter notre approche à des situations où l'utilité de l'occurrence d'un motif dans une transaction est le produit des utilités des items au sein de la transaction. En perspective, nous voudrions étendre notre approche à d'autres structures de données complexes telles que les séquences et les graphes.

# Références

Agrawal, R. et R. Srikant (1994). Fast algorithms for mining association rules in large databases. VLDB '94, pp. 487–499. Morgan Kaufmann Publishers Inc.

Ahmed, C. F., S. K. Tanbeer, B. Jeong, et Y. Lee (2009). Efficient tree structures for high utility pattern mining in incremental databases. *IEEE Transactions on Knowledge and Data Engineering 21*(12), 1708–1721.

Boley, M., S. Moens, et T. Gärtner (2012). Linear space direct pattern sampling using coupling from the past. In *Proceedings of the 18th ACM SIGKDD*, pp. 69–77. ACM.

Diop, L. (2020). *Echantillonnage sous contraintes de motifs Structurés*. Ph. D. thesis, Gaston Berger University, Saint-Louis, Senegal.

Diop, L., C. T. Diop, A. Giacometti, et A. Soulet (2022). Pattern on demand in transactional distributed databases. *Information Systems 104*, 101908.

Lin, J. C.-W., T. Li, P. Fournier-Viger, T.-P. Hong, J. Zhan, et M. Voznak (2016). An efficient algorithm to mine high average-utility itemsets. *Advanced Engineering Informatics 30*(2), 233 – 243.

Shie, B.-E., P. S. Yu, et V. S. Tseng (2013). Mining interesting user behavior patterns in mobile commerce environments. *Applied Intelligence 38*(3), 418–435.

Singh, K., S. S. Singh, A. Kumar, et B. Biswas (2019). Tkeh : an efficient algorithm for mining top-k high utility itemsets. *Applied Intelligence 49*(3), 1078–1097.

Song, W., C. Zheng, C. Huang, et L. Liu (2021). Heuristically mining the top-k high-utility itemsets with cross-entropy optimization. *Applied Intelligence*, 1–16.

van Leeuwen, M. (2014). *Interactive Data Exploration Using Pattern Mining*, pp. 169–182. Berlin, Heidelberg : Springer Berlin Heidelberg.

Yao, H., H. J. Hamilton, et C. J. Butz (2004). A foundational approach to mining itemset utilities from databases. In *Proceedings of the Third SIAM International Conference on Data Mining*, pp. 482–486.

# Summary

High-Utility Itemset mining algorithms are methods for discovering relevant information in a database where items are weighted. Their usefulness has been widely demonstrated in many real world applications. We propose an algorithm to sample itemsets with high average utility under length constraints to avoid the long and rare itemsets with items having low weights.