République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Saad Dahlab Blida

N° D'ordre :.....

Faculté des sciences

Département d'informatique

Mémoire Présenté par :

BENATTOU Ibtissem

LARBI Khalida

En vue d'obtenir le diplôme de Master

Domaine : Mathématique et informatique

Filière : Informatique
Spécialité : Informatique
Option : Ingénier logicielle

Thème

Extraction des itemsets fréquents à partir des données imparfaites

Soutenu le :

M.	Farah	Président
M.	Gessem	Examinateur
Mme	ZAHRA FATMA ZOHRA	Promotrice

Promotion : 2016 / 2017

Remerciements

C'est avec l'aide de Dieu que ce travail a vu le jour,

Il n'aurait pu être achevé sans le soutien, les conseils et les encouragements de certaines personnes auxquelles nous tenons à exprimer ici nos sincères remerciements.

En premier lieu, nous exprimons toute notre gratitude pour notre Promotrice, Mlle. ZAHRA pour ces précieux conseils, sa disponibilité, la confiance qu'elle nous a toujours témoigné et la sollicitude dont elle nous a entouré, et ce tout au long de l'élaboration du présent travail. Nous n'oublions pas non plus nos Enseignants, qui nous ont transmis leur savoir tout au long du cycle d'études à l'UNIVERSITE DE BLIDA 1.

Nous adressons une pensée particulièrement affective à nos Amis qui ont rendu agréables nos longues années d'études.

Nous tenons enfin à remercier tous ceux qui ont collaborés de près ou de loin à l'élaboration de ce travail. Qu'ils acceptent nos humbles remerciements.

Dédicaces

Que ce travail témoigne de mes respects :

A mes parents :
Ma chère mère CHERIFA, pour ses sacrifices
Depuis qu'elle m'a mis au monde, et qui m'a aidé surtout dans les Moments difficiles
Mon père MOHAMMED, qui m'a toujours soutenu et guidé pour affronter les difficultés de la vie,
A mes sœurs :
Grâce à leurs tendres encouragements et leurs grands sacrifices, ils ont pu créer le climat affectueux et propice à la poursuite de mes études.
Aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux.
Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi
A mes frères :
Qui ont toujours été présents à mes côtés
Ils vont trouver ici l'expression de mes sentiments de respect et de reconnaissance pour le soutien qu'ils n'ont cessé de me porter.
A mon binôme Khalida et sa famille ainsi que mes amies intimes Hayet, Kawther, Sarah, Faiza , Zahra, Nassima.
A ma promotrice Zahra qui m'aide avec son grand encouragement pour compléter mon travail.
Ils vont trouver ici le témoignage d'une fidélité et d'une amitié infinie.
Enfin à tous mes amis que j'ai n'ai pas cités, à tous ceux qui me connaissent.
Tous ceux que j'aime....

DEDICACE

Je dédie ce mémoire

A mes chers parents qui représentent tout mon monde mon père et ma mère pour leur patience, leur amour, leurs soutiens et leurs encouragements.

A mes sœurs Fatma zohra, Yamina, Fidia et mon frère Fouad je vous aime

A mes nièces et neveux qui nous combles de joie Rayan, Raouf, Lilia, Adam, Nouredine, Alaa, Maria je vous adore.

A mes beaux-frères Tarek, Abdelaziz et Mhamed.

A mon binôme Ibtissem ainsi que toute sa famille.

A tous mes amis qui m'ont rendus heureuse et spécialement, Nassima merci pour ton aide, Imen, Khadidja, Nour et ceux que j'ai connu durant ces cinq dernière années.

A ma promotrice qui a cru en moi et ma soutenue merci pour votre aide et votre gentillesse ainsi que pour votre merveilleux encadrement.

A toute ma famille

A tous ceux que j'aime . . .

Khalida

RESUME

Résumé

L'extraction des itemsets fréquents (EIF) à partir des données constitue l'étape cœur de plusieurs méthodes de fouille de données. Cependant, la majorité des techniques d'EIF ne prennent pas en considération l'aspect imparfait. Ce problème d'imperfection touche en effet plusieurs systèmes d'informations. Il est généré dans la phase d'acquisition des données et peut être sous plusieurs formes tel que l'incertitude l'incomplétude et l'imprécision on s'intéresse dans notre travail à ce genre de données. En effet, notre travail consiste à proposer une méthode d'extraction des itemsets fréquents à partir des données imparfaites Plus précisément, la méthode doit prendre en considération les différents types d'imperfection de données citées préalablement.

Mots clés : Extraction des itemsets fréquents, données imparfaites.

Abstract

The extraction of frequent itemsets (EIFs) from the data is the cornerstone of several data mining methods. However, the majority of EIF techniques do not take into account the imperfect aspect. This problem of imperfection affects several information systems. It is generated in the data acquisition phase and can be in several forms such as uncertainty incompleteness and vagueness are interested in our work at this kind of data. Therefore, our work consists in proposing a method for extracting frequent itemsets from the imperfect data. More precisely, the method must take into account the different types of imperfection of previously cited data.

Keywords: Extraction of frequent itemsets, imperfect data.

RESUME

ملخص

استخراج الأنماط المتكررة من البيانات يشكل خطوة المركزية للعديد من أساليب استخراج البيانات. ومع ذلك، فإن معظم تقنيات المستعملة لا تأخذ بعين الاعتبار مظهر الكمال. هذه مشكلة النقص وتؤثر على عدة نظم المعلومات. يتم إنشاؤها في مرحلة الحصول على البيانات ويمكن أن تكون في أشكال عديدة مثل عدم اكتمال عدم اليقين وعدم دقة الاهتمام في عملنا لهذه البيانات. ولذلك، مهمتنا هي اقتراح طريقة استخراج الأنماط المتكررة من البيانات الكمال على وجه التحديد، ينبغي أن تأخذ بعين الاعتبار طريقة أنواع مختلفة من البيانات الناقصة استشهد في وقت سابق.

كلمات البحث: استخراج الأنماط المتكررة، البيانات الناقصة.

SOMMAIRE

SOMMAIRE

CHAPITRE 03 : Extraction des itemsets fréquents à partir de données imparfaites

CHAPITRE 04 : Approche proposée

SOMMAIRE

CHAPITRE 05 : Teste et validation

Conclusion générale

Liste des abréviations

EIF : Extraction des Itemsets Fréquents.

ECD : Extraction des Connaissance à partir des Données.

DSET: Dempster Shafer Evidential Théory.

bba : La masse de croyance élémentaire.

Foc : Elément focaux.

BoE : Corps d'évidence.

Bel : La fonction de croyance.

Pl : La fonction de plausibilité.

SuppA : Le support d'un sous ensemble flou A.

hA : La hauteur du sous ensemble flou A.

NoyA : Le noyau d'un sous ensemble flou A.

EQUp : Classe d'équivalence.

$\underline{appr}$: Approximation inferieur.

$\overline{appr}$: Approximation supérieure.

FP-growth : Frequent Pattern growth.

FP-Tree : Fréquent Pattern tree.

expSup : Expected support.

BIT: Belief Itemset Tree.

FIM : Frequent Itemset Maintenance .

EDB : Evidentiel Data Base .

ER-Apriori : Evidential Reliable Apriori.

FBD : fuzzy base de données.

Listes des figures

Liste des tableaux

Introduction générale

1. Contexte

L'extraction des itemsets fréquents (EIF) à partir de données a reçu une attention particulière de la part des chercheurs ces derniers temps puisqu'elle constitue l'étape cœur de plusieurs méthodes de fouille de données à l'instar de l'extraction des règles d'associations, la construction de certains classifieurs, l'extraction de séries temporelles et d'autres techniques de fouille de motifs (dite aussi patterns).

Néanmoins, la majorité des techniques d'EIF ne prennent pas en considération l'aspect imparfait des données générées par les applications du monde réel. Ce problème d'imperfection touche en effet plusieurs systèmes d'informations. En sciences expérimentales, les chercheurs se basent sur des expérimentations dont les données générées sont sauvegardées et par la suite traitées. Ces valeurs observées ou mesurées sont la plupart du temps imprécises ou incertaines et ou incomplètes. En médecine, les docteurs se trouvent souvent dans l'obligation d'émettre un diagnostic en présence de symptômes imprécis voire incertains et même incomplet. Les données générées par certains systèmes à base de capteurs sont aussi imparfaites. Les capteurs d'un système unique peuvent produire des informations à différents niveaux de confiance. En plus, chacun d'eux peut produire une information incertaine, imprécise ou incomplète.

2. Problématique

Le problème de données imparfaites (manquantes, imprécises, incertaines, ...) est un problème connu dans le domaine de la fouille de données et de l'apprentissage automatique où, dans la base d'apprentissage, on rencontre des objets ayant des valeurs manquantes et/ imprécises pour certains attributs. Cela arrive pendant la phase d'acquisition des données du processus de l'ECD. Prendre une décision en présence de données imparfaites est une tâche difficile. Les natures de cette imperfection des connaissances sont :

1. Les incertitudes concernant un doute sur la validité d'une connaissance.
2. Les imprécisions correspondent à une difficulté dans l'énoncé de la connaissance.
3. Les incomplétudes sont des absences de connaissances ou des connaissances partielles sur certaines caractéristiques du système.

La question qui se pose, alors, est " Comment modéliser ces imperfections, et comment les utiliser pour l'extraction des connaissances de façon automatique ?", et plus précisément dans le cadre de l'extraction de motifs fréquents.

3. Objectifs

L'extraction des motifs fréquents (EIF) est une technique utilisée en fouille de données, elle s'appuie sur des principes relativement simples. Son objectif est de trouver les motifs qui apparaissent fréquemment dans un ensemble de données.

En effet, L'objectif de ce travail est de développer une méthode d'extraction de motifs fréquents à partir des données imparfaites. Plus précisément, la méthode doit prendre en considération les différents types d'imperfection de données (incomplétude, imprécision, incertitude).

4. Organisation du mémoire

Le mémoire se réparti en cinq chapitres.

Chapitre 1 : « Données imparfaites »

Ce chapitre est consacré à la présentation des données imparfaite (incomplète, incertaine, imprécise) aux différents types d'imperfection et leurs sources.

Chapitre 2 : « Extraction des itemsets fréquents à partir de données parfaites »

Dans ce chapitre nous allons présenter les différents algorithmes d'extraction à partir des données incertaines, et imprécises.

Chapitre 3 : « L'extraction des itemset fréquents à partir de données imparfaites »

Dans ce chapitre nous détaillerons l'implication de certaines théories mathématiques pour l'extraction d'itemset fréquents.

Chapitre 4 : « Approche proposée »

Ce chapitre, sera réservé pour présenter notre solution proposée et évaluation de notre application.

Chapitre 5 : « Test et Validation»

Le dernier chapitre concrétise et valide la méthode adoptée.

Chapitre 1 :

«Données imparfaites »

1. Introduction

Vu l'habilité et le comportement naturel de l'être humain et sa capacité de prendre des décisions en présence d'une grande masse de données souvent imparfaites (incomplètes, incertaines ou imprécises...), l'automatisation du traitement de ce type de données à fait l'objet de plusieurs recherches dont l'objectif était de proposer des modèles et méthodes pour représenter et manipuler ce type de données pour faire face à des situations réelles.

2. Définition de la donnée imparfaite

A la rencontre d'un ensemble de données imparfaites, la première chose à faire est de donner un "sens" à ces données pour pouvoir les interpréter. Si les données dans le monde réel sont incomplètes ou incertaines, plusieurs scénarios et états sont possibles, mais on ne peut pas dire lequel de ces scenario représente l'état du monde réel. Donc une base de données qui contient des données incomplètes et / ou incertaines représente implicitement un ensemble d'états possibles et une description proche de la réalité.

Selon Vaughn et al **[01]** La notion d'imperfection dans les données fait appel à quatre Concepts : l'imprécision, l'incertitude, l'incomplétude et l'erreur. Illustrée sur la Figure 1.1.

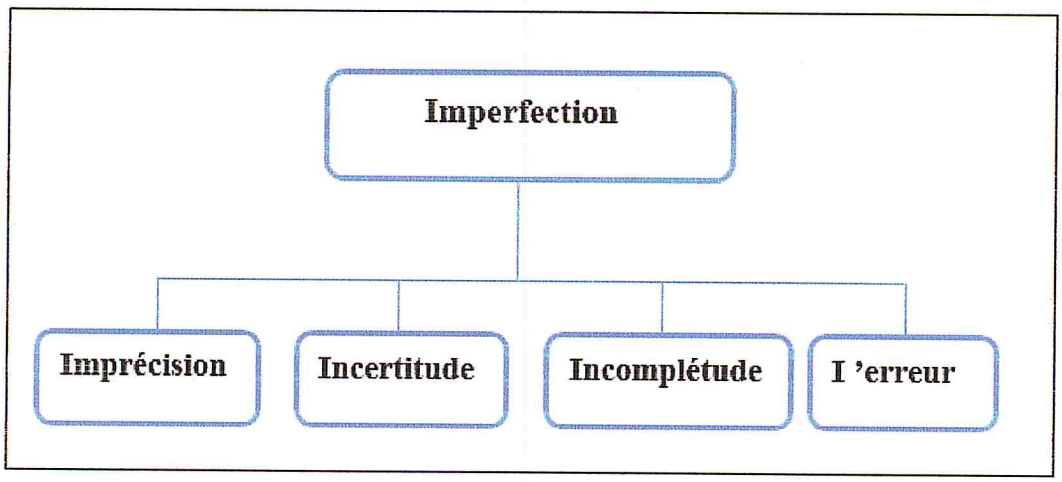


Figure 1.1 : Grands types d'imperfection. **[01]**

Les quatre concepts d'imperfection sont définis comme ci-dessous :

2.1. Imprécision : Elle correspond à une difficulté dans l'énoncé de la donnée, soit parce que des :

- Données numériques sont mal connues (causé par l'insuffisance des instruments d'observation, d'erreurs de mesure (poids à 1% près) ou encore de connaissances flexibles (la taille d'un adulte est environ entre 1.50 et 2 mètres)).
- Des termes du langage naturel sont utilisés pour qualifier une caractéristique du système de façon vague. (provient de l'interprétation spontanée de la donnée à titre d'exemple (température douce, grand appartement, proche de la plage) ou de l'utilisation de catégories aux limites mal définies (enfant, adulte, vieillard)).

Remarque : La donnée imprécise dénote un ensemble de valeurs possibles et la valeur réelle est l'un des éléments de cet ensemble de valeurs. Donc, elle n'est pas incorrecte et ne compromet pas l'intégrité des données.

2.2. Incertitude : Parfois notre connaissance du réel ne peut pas être énoncée avec confiance ou garantie absolue. Car la donnée énoncée avec l'incertitude n'est pas incorrecte et ne compromet pas son intégrité. Bien que : l'âge d'un personne de 20 ou 24 ans est imprécis. La donnée : l'âge est probablement 20 est une donnée incertaine, dans quelques cas, le degré de la certitude est donnée : l'âge est 32 ans avec une probabilité de 0.6 et 33 avec une probabilité de 0.4.

Remarque : L'imprécision et l'incertitude sont deux notions très liées, on peut dans quelque cas modéliser l'imprécision par l'incertitude et vice versa. Plus la donnée est précise plus elle est certaine à titre d'exemple «je suis sûr que la note est entre 10 et 12 mais je ne suis pas certaine qu'elle soit 11 » ou bien « je suis certaine que je serais à l'université l'après-midi, mais je ne suis pas sûr que je serais la à 13h30min » si la valeur précise mais pas certaine est entourée par d'autres valeurs possibles ceci incrémente la certitude mais l'imprécision sera importante également.

2.3. Incomplétude : La donnée est dite incomplète si elle contient au moins une valeur manquante, dans ce cas on a uniquement une donnée partielle du réel perçu. Les incomplétudes sont :

- Des absences de données ou des données partielles sur certaines caractéristiques de l'objet. Elles peuvent être dues à l'impossibilité d'obtenir certains renseignements (fichiers de malades dans lesquels certaines rubriques ne sont parfois pas remplies) ou à un problème au moment de la capture de la donnée.
- Elles peuvent aussi être associées à l'existence de données générales sur l'état d'un système, habituellement vraies, soumises à des exceptions que l'on ne peut pas énumérer ou prévoir, selon les cas, (" généralement, Pierre est à son bureau tous les jours ", sauf s'il est malade ou si un événement grave survient dans sa famille).
- Elles sont généralement liées à l'existence de données implicites, par exemple dans une recherche d'information auprès d'experts. Ces imperfection ne sont pas exclusives l'une de l'autre et l'incomplétude est toujours ramenée à l'imprécision.

2.4. L'erreur :

C'est le type le plus simple de donnée imparfaite. La donnée Stockée est incorrecte quand elle est différente de l'information vraie. Cependant d'après un balayage de la littérature il n'existe pas de représentation ou une modélisation précise pour les données erronées sauf que ce sont des données aberrantes.

3. Les causes d'imperfection des données

Cette imperfection des données est due à plusieurs raisons selon Ben Amor Sarah et Jean-Marc Martel **[02]** on en cite deux principales causes :

- L'obtention des données à partir du réel s'effectue en deux étapes : l'observation et la représentation. La première se produit à travers des intermédiaires (humains) qui sont généralement soumis à des erreurs, des imprécisions et des incertitudes. La seconde étape est celle de la représentation de ces données. Autant l'observation que la représentation entraîne une perte de données est d'autant plus grande que le système est complexe.
- L'absence de rigueur ou de flexibilité inhérente au système lui-même et son fonctionnement, c'est le cas pour toutes les caractéristiques de phénomènes naturels tel

que la durée de maturation d'un fruit, la taille d'un animal adulte, le passage progressif et non strict du jours a la nuit .

C'est aussi le cas de certains systèmes artificiels tels que la charge maximale d'un ascenseur indiquée en kilogrammes dans un souci de simplicité mais à laquelle on peut ajouter quelques grammes sans problèmes majeur où le nombre maximal de voyageurs que peut contenir un wagon de métro, dépendant du degré de compression accepté par les passagers.

4. Les théories qui traitent les aspects d'imperfections des données

La figure 1.2 si dessous nous montre la modélisation des déférents types d'imperfection de données selon les théories. En effet le problème d'incertitude de données peut être traité par la théorie des probabilités de part l''incertitude on trouve aussi l'ambiguïté. Nous avons aussi le problème d'imprécision qui peut être traité par la théorie des ensembles flous.il reste aussi le problème des données incomplète qui peut être traité par la théorie des possibilités enfin le problème de granularités des données peut être traité pas la théorie des ensembles d'approximation ou bien rough set.

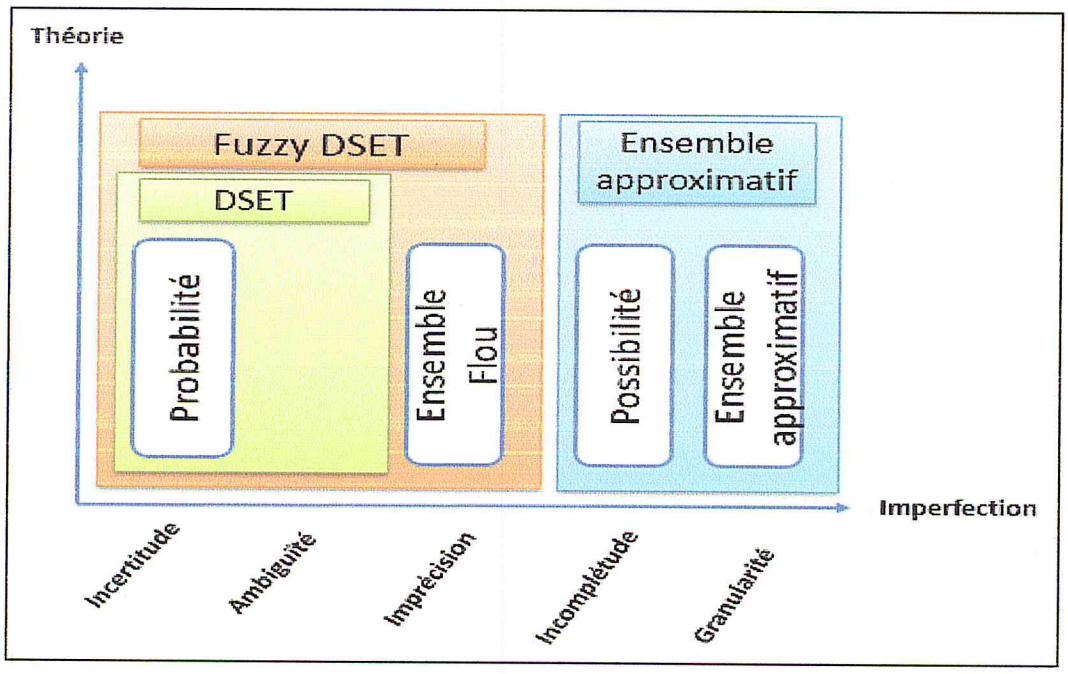


Figure 1.2 : Les imperfections en fonction des théories. **[01]**

Cette figure nous montre comment sont distribuées les imperfections en fonction des théories qui les traitent.on remarque que deux grandes theories

4.1. La Théorie des fonctions de croyance : Dempster Shafer Evidential théory (DSET)

Appelée aussi théorie de l'évidence, a été introduite en 1967 par Dempster [3] et formalisée mathématiquement par Shafer. Nous présentons dans cette section les concepts formels de cette théorie.

Soit $\Theta = \{\theta 1, \theta 2, \ldots, \theta n\}$ un ensemble fini non-vide incluant n hypothèses, Le référentiel de définition utilisé pour évaluer la véracité d'une proposition est constitué de tous les sous-ensembles possibles de Θ, soit l'ensemble 2^{Θ}. La fonction de masse élémentaire m d'une hypothèse$X \subseteq \Theta$, notée $m(X)$, représente la partie du degré de croyance placée sur X et qui n'a pas été distribuée aux sous-ensembles de X.

La masse de croyance élémentaire, notée bba est définie par : $m: 2\Theta \rightarrow [0,1]$.

m Satisfait deux conditions qui trouvent leur correspondance dans la théorie des probabilités à savoir :

- La masse de l'ensemble vide est nulle: $m(\emptyset) = 0$.
- La somme des masses de toutes les hypothèses correspond à l'unité :

$$\sum_{j=0}^{d} m(X) = 1. \quad (1)$$

Les sous-ensembles de Θ sur lesquels sont placées des masses strictement positives sont appelés **éléments focaux,** leur ensemble est noté Foc. La paire $\{Foc, m\}$ est appelé **corps d'évidence** noté BoE. La fonction de croyance bel est définie à partir de la fonction de masse$m$, dans la mesure où la croyance d'un événement $A \subseteq \theta$ reflète la croyance totale assignée à A, c-à-d, la somme des masses de tous les sous-ensembles de A .

$$bel(A) = \sum_{B \subseteq A} m(B). \quad (2)$$

D'autre part, $\boldsymbol{Pl(A)}$est la fonction de plausibilité et est définie comme suit :

$$\mathrm{Pl(A)} = \sum_{B \cap A \neq 0} \mathrm{m(B)}. \quad (3)$$

La *fonction de plausibilité (pl)* est la fonction duale de la fonction de croyance, elle mesure. L'intensité avec laquelle une proposition A peut être considérée comme vraie. La plausibilité de A est donc la somme des masses de tous les éléments focaux qui sont compatibles avec A.

Dans ce cas, le produit cartésien permet la combinaison. Que ce soit deux croyances fonction $\boldsymbol{m1}$ et $\boldsymbol{m2}$ sont définis respectivement dans les $\boldsymbol{\theta 1}$ et $\boldsymbol{\theta 2}$, le produit cartésien est exprimé comme suit :

$$\mathrm{m}^{\theta}_{1\times 2}(\mathrm{A}\times\mathrm{B}) = \mathrm{m}^{\theta 1}_{1}(\mathrm{A})\times\mathrm{m}^{\theta 2}_{2}(\mathrm{B}). \quad (4)$$

D'après Dempster **[10]** et Smets **[29]** la probabilité pianistique est introduite permettant décision probabiliste de BBA suivant cette formule :

$$\mathrm{BetP(H_n)} = \sum_{\mathrm{A}\subseteq\theta} \frac{|\mathrm{H_n}\cap\mathrm{A}|}{|\mathrm{A}|}\times\mathrm{m(A)} \qquad \forall \mathrm{H_n} \in \theta. \quad (5)$$

Où $|\cdot|$ est l'opérateur de cardinalité.

4.2. Ensembles flous et d'approximation

Il existe deux extensions à la théorie des ensembles : les ensembles flous (fuzzy sets) et ensembles d'approximation (Rought sets) permettant de modéliser les données imprécises et incertaines et aussi incomplète.

4.2.1. Théorie des ensembles flous

La théorie des ensembles flous est en fait selon Zadeh **[04]** un pas vers un rapprochement entre la précision des mathématiques classiques et la subtile imprécision des données contenues dans le monde réel .Les domaines d'application dans lesquels il existe des utilisations de la logique floue sont très variés : médecine, biologie…etc.

Dans ce qui va suivre, nous allons présenter brièvement les concepts de base et les principes de l'arithmétique de la théorie des ensembles flous **[04].**

- **Définition d'un sous-ensemble flou**

Dans la théorie des ensembles classiques, il n'y a que deux situations acceptables pour un élément, appartenir ou ne pas appartenir à un sous-ensemble. Le mérite de Zadeh **[05]** a été de tenter de sortir de cette logique booléenne en introduisant la notion d'appartenance pondérée permettre des graduations dans l'appartenance d'un élément à un sous-ensemble, c'est-à-dire d'autoriser un élément à appartenir plus moins fortement à ce sous-ensemble. Soit X un ensemble de référence et soit x un élément quelconque de X. Un sous-ensemble flou A de X est défini comme l'ensemble des couples :

$$A = \{(x, \mu A(x)), x \in X\} \text{ avec} \qquad (6)$$

$$\mu A : X \rightarrow [0,1]$$

Ainsi, un sous-ensemble flou A de X est caractérisé par une fonction d'appartenance $\mu A(x)$ qui associe, à chaque point x de X un réel dans l'intervalle [0,1] $\mu A(x)$ représente le degré d'appartenance de x à A. On observe les trois cas possibles suivants : **[05]**

$$\begin{cases} \mu A(x) = 0 \\ 0 < \mu A(x) < 1 \\ \mu A(x) = 1 \end{cases} \qquad (7)$$

Où, $\mu A(x) = 0$ si x n'appartient pas A, $0 < \mu A(x) < 1$ si x appartient partiellement à A, et $\mu A(x) = 1$ si x appartient entièrement à A.

On peut faire remarquer que si A est un sous-ensemble classique, la fonction d'appartenance qui lui est associée ne peut prendre que les valeurs extrêmes 0 ou 1.

On a dans ce cas :

$$\mu A(x) = \begin{cases} 0, si\ x \notin A \\ 1, si\ x \in A \end{cases} \qquad (8)$$

- **Caractéristiques d'un sous-ensemble flou**

Un sous-ensemble flou est complètement défini par la donnée de sa fonction d'appartenance. A partir d'une telle fonction, un certain nombre de caractéristiques du sous-ensemble flou peuvent être étudiées. **[05]**

- **Support et Hauteur**

Ces deux caractéristiques, pour l'essentiel montrent, dans quelle mesure un sous-ensemble flou A de X diffère d'un sous-ensemble classique de X.

La première est le support et la deuxième la hauteur. Le support d'un sous-ensemble flou de A de X, noté ***Supp A ( ),*** est l'ensemble de tous les éléments qui lui appartiennent au moins un petit peu. Formellement :

$$Supp(A) = \{x \in X \mid \mu A(x) > 0\} \qquad (9)$$

La hauteur du sous-ensemble flou A de X, notée ***hA ( ),*** est le plus fort degré avec lequel un élément de X appartient à A. Formellement :

$$h(A) = \sup \mu(x) \; tel \; que \; x \in X \qquad (10)$$

- **Noyau**

Un sous-ensemble flou est normalisé si sa hauteur ***hA ( ) = 1***. Le noyau d'un sous-ensemble flou A de X, noté ***Noy A ( )***, est l'ensemble de tous les éléments qui lui appartiennent totalement (avec un degré 1). **[07]**

$$Noy(A) = \{x \in X \mid \mu A(x) = 1\} \qquad (11)$$

- **Cardinalité**

La cardinalité d'un sous-ensemble flou A de X, noté |A|, est le nombre d'éléments appartenant à A pondéré par leur degré d'appartenance. Formellement, pour A fini

$$|A| = \sum \mu A(x) \; tel \; que \; x \in X \qquad (12)$$

Si A est un sous-ensemble ordinaire de X, sa cardinalité est le nombre d'éléments qui le composent.

4.2.2. Théorie des ensembles d'approximation (rough sets)

La théorie des ensembles approximatifs a été introduite par Zdzislaw Pawlak en 1982 **[06]** est une approche mathématique qui traite les données manquantes, imprécises et incertaines. Peut être considérée comme un nouvel outil mathématique pour l'analyse de données imparfaites. La théorie a trouvé des applications dans beaucoup de domaines, tels que l'aide à la décision, opérations bancaires, médecine…etc. Dans ce qui va suivre, nous allons et selon **[06]** présenter brièvement les concepts de base et les principes de l'arithmétique de la théorie des ensembles d'approximation.

- **Systèmes d'information et relation d'équivalence**

Un système d'information est représenté par un tableau de données. L'étude d'un système d'information par la théorie des ensembles approximatifs est définit formellement par la paire $I= (U, A)$ où U est un ensemble fini non vide d'objets appelés univers et A est un ensemble fini non vide d'attributs tels que :

$fa : U \rightarrow Va$ .Pour chaque a ∈ A La valeur discrète non vide V_a s'appelle le domaine de a. La théorie des ensembles approximative originale traite les systèmes d'information complète dans lesquels $\forall \in U, a \in A, fa(x)$ Est une valeur précise.

N'importe quel système d'information prenant la forme $I = (U, A \cup \{d\})$ s'appelle une table de décision où d ∉ A s'appelle une décision et des éléments de A s'appellent les conditions. Prenons $V_d = \{d_1 \ldots . d_k\}$ dénotant l'ensemble de valeurs de l'attribut de décision, la décision d détermine un ensemble $\{C_1, C_2, \ldots, C_k\}$ d'univers U des hypothèses où $Ci = \{x \in U | fd(x) = di\}$. $1 \leq i \leq k$ S'appelle la classe ou le concept de décision sur U. Nous supposons que chaque objet dans U a une certaine valeur de décision dans V_d.

Un exemple de table de décision complète est montré dans Le tableau 1.1 L'univers est $U=\{x_1, x2, \ldots . x_8\}$, l'ensemble de condition est $A= \{Température, mal\ de\ tête, doleur\}$ et la décision d est $grippé$. Dans ce tableau x_1, x_4 et x_5 ont exactement les mêmes valeurs sur l'attribut $P = \{Température, mal\ de\ tête\}$ Ce cas est (par paire) indiscernable qui utilise les attributs disponibles. Basé sur les relations d'équivalence/indiscernable entre les objets, les classes d'équivalence de tous les objets sur P sont :$\{x_1, x_4, x_5\}$, $\{x_2\}$ $\{x_3\}$ $\{x_6, x_8\}$ et $\{x_7\}$.

Formellement, dans des systèmes d'information, la relation devient :

$$EQUp(x,y), p \subset A \qquad (13)$$

Dénote une relation binaire entre les objets qui sont équivalents en termes de valeurs des attributs dans P. La relation d'équivalence est réflexive, symétrique, et transitive. Représenter par :

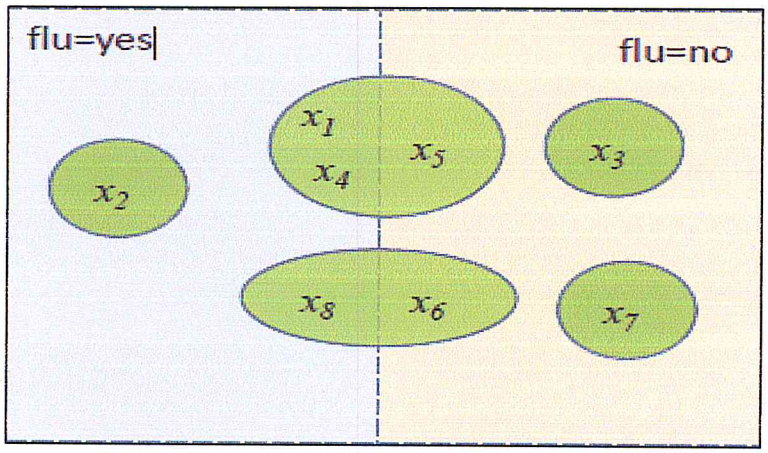
$$E_p(X)= \{y \in U | EQU_p(y,x)\} \quad (14)$$

Soit l'ensemble de tous les objets qui sont équivalents à x par P, qui s'appelle alors une classe d'équivalence. La famille de toutes les classes d'équivalence sur U basé sur la relation d'équivalence par catégories et est dénotée par

$$U/EQU_p. \qquad (15)$$

Tableau 1.1 : tableau de décision complet. [08]

Cases	Patients			Décision
	température	Mal de tête	Nausée	
X1	Haut	Oui	Non	Oui
X2	Très haut	Oui	Oui	Oui
X3	Haut	Non	Non	Non
X4	Haut	Oui	Oui	Oui
X5	Haut	Oui	Oui	Non
X6	Normale	Oui	Non	Non
X7	Normale	Non	Oui	Non
X8	Normale	Oui	Non	Oui



Le schéma 1.1 : les ensembles d'approximations. **[08]**

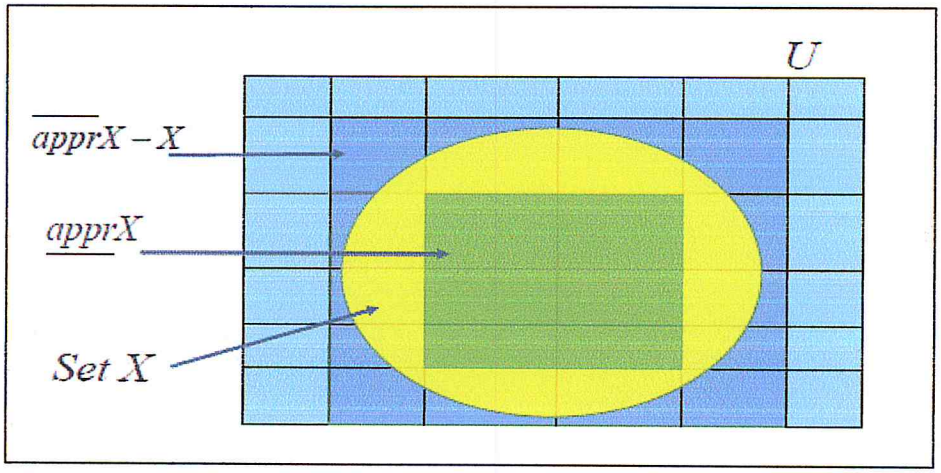
Le schéma 1.1 explique l'approximation de l'ensemble des patients utilisant deux attributs conditionnels Température et mal de tête.

- **L'espace d'approximation**

Supposons que nous devons décrire un groupe de patients $X \subseteq U$, qui sont dans le tableau 1.1 en employant le sous-ensemble conditionnel d'attribut P qui se compose de la ***température*** et du ***mal de tête***. Du schéma 1.1 X ne peut pas être exactement décrit en termes de ***P*** parce que l'ensemble peut inclure ou exclure les objets qui sont différent sur la base des attributs de P par exemple, il n'y a aucune manière de représenter X par un seul ensemble.

$$\{x \in U | f_{Temperature}(x) = high \wedge f_{mal\ de\ tête}(x) = yes\} \quad \textbf{(16)}$$

C'est parce que x1 et x5 sont équivalents entre eux mais x5 dans le concept de la $grippe = no$. Cependant, peut être rapproché.



Le **schéma 1.2 :** illustration des approximations. **[08]**

Représentation schématisée des ensembles approximatifs inférieurs et supérieurs de la théorie des ensembles approximatifs.

A partir des classes d'équivalences Pawlak **[07][08]** a défini un espace d'approximation qui contient les approximations inférieures et supérieures indiquées par $\underline{appr_p}X$ et $\overline{appr_p}X$, respectivement, de l'ensemble $X \subseteq U$ comme suit :

$$\begin{aligned} apprp\ X &= \bigcup\{Ep(x)|x \in U, Ep \subseteq X\} \\ &= \{x \in U|Ep(x) \subseteq X\} \\ apprp\ X &= \bigcup\{Ep(x)|x \in U, Ep(x) \cap X \neq \emptyset\} \\ &= \{x \in U|Ep(x) \cap X \neq \emptyset\} \end{aligned} \quad (17)$$

Définir boundpX= $\underline{appr_p}X$- $\overline{appr_p}X$, est appelé la région limite de X. L'ensemble de U- $\overline{appr_p}X$ s'appelle la région extérieure du X. X est dit *Rough* si la région limite de X est *none-vide* d'une part, placer X est croquant si la frontière de X est vide. Le schéma 1.2 montre l'univers U, l'objet X réglé et les approximations du X. Du système d'information montré dans le tableau 1.1 nous avons $X=\{x \in U|\ f_{grippe}(x)=oui\}$ Et $P = \{Température, mal\ de\ tête\}$.

On peut déduire :

- ✓ L'approximation inferieure $\underline{appr_p}X=\{x_2\}$,
- ✓ l'approximation supérieure $\overline{appr_p}X=\{x_1, x_2, x_4, x_5, x_6, x_8\}$.
- ✓ La région limite $bound_pX=\{x_1, x_4, x_5, x_6, x_8\}$.
- ✓ U - $\overline{appr_p}X=\{x_3, x_7\}$.

- **L'ensemble approximatif (Rough Set)**

Les couple $\underline{appr_p}X$ et $\overline{appr_p}X$ composé d'approximations inférieure et supérieure s'appelle un ensemble approximatif (Rough Sets), la représentation de l'ensemble X peut être donnée comme

$$\alpha_p(x) = \frac{|apprpX|}{|apprpx|}. \quad \textbf{(18)}$$

Là où $X \neq \emptyset$, |Y| dénote la cardinalité d'un ensemble Y.

La représentation approximative de l'ensemble X est le rapport du nombre d'objet complètement dans X et du nombre d'objets appartenant probablement a X.

$\emptyset \subseteq$ apprp X $\subseteq$ apprp X, nous avons $0 \leq \alpha_{p(X)} \leq 1$ **(19)**

La formule (18) représente une mesure de comment l'ensemble approximatif rapproche l'ensemble X.

L'ensemble X est rough en ce qui concerne P si les approximations inférieures et supérieures de X sont égales. La région limite est vide et $\propto_p(X)=1$. D'une part, si la région limite n'est pas vide Alors $\propto_p(X) < 1$.

A partir des approximations inférieures et supérieures d'un ensemble, l'ensemble approximatif peut être classé par catégorie comme classes de base suivantes des ensembles approximatifs.

Quatre catégories :

- L'ensemble X est *approximativement définissable* si $\underline{appr_P}\ X \neq \emptyset$ et $\overline{appr_P}\ X \neq U$.

Ceci signifie cela sur le jeu d'attributs P, il y a des objets dont nous pouvons être certains de leurs appartenance à X, et il y a également des objets que nous pouvons

définitivement exclure de l'ensemble X. **[08]**

• L'ensemble X est *définissable à l'intérieur* si $\underline{appr_P}\ X \neq \emptyset$ et $\underline{appr_P} X = U$. Ceci signifie que sur le jeu d'attribut P, il y a des objets dont nous pouvons être certains de leurs appartenances à X, mais il n'y a aucun objet que nous pouvons définitivement exclure de l'ensemble X. **[08]**

• L'ensemble X est *définissable de l'extérieur* si $\underline{appr_P}\ X = \emptyset$ et $\overline{appr_P} X \neq U$. Ceci signifie que sur le jeu d'attribut P, il n'y a aucun objet auquel nous pouvons être certains de leurs appartenances à X, mais il y a des objets que nous pouvons définitivement exclure de l'ensemble X. **[08]**

• L'ensemble X est *totalement non définissable* si $\underline{appr_P}\ X = \emptyset$ et $\overline{appr_P}\ X = U$. Ceci signifie cela sur le jeu d'attribut P, il n'ay a aucun objet auquel nous pouvons être certains de son appartenance a X, et il n'y a aucun objet que nous pouvons définitivement exclure de l'ensemble X. Ainsi, sur le jeu d'attribut P, nous ne pouvons pas décider si n'importe quel objet est, ou n'est pas, un membre de X. **[08]**

Dans le tableau1.1, l'ensemble $X=\{x \in U | f_{flu}(x)= yes\}$ est rough en raison de $\underline{appr_P}\ X \neq \emptyset$ et l' $\overline{appr_P}\ X \neq U$. **[08]**

5. Conclusion

Dans cette partie, nous avons donné quelques notions et définitions relatives aux données imparfaite (incomplète, incertaine, incomplète) et les différents types d'imperfection ainsi que les causes de ce problème avec des exemples et les modèles mathématiques qui permettent de modéliser ce type de données. Dans le chapitre suivant on explique les algorithmes pour l'extraction des itemsets fréquent qui utilisant ces model pour représente les données.

Chapitre 2 :

« L'extraction des itemset fréquents des données parfaites »

1. Introduction

De nos jours, l'extraction des motifs fréquents est nécessaire également sur des contextes avec la probabilité des données c'est dû à plusieurs techniques de rassemblement de données. En outre, les applications courantes également ont augmenté les données avec des imperfections causées par les mesures imprécises, incertaines ou erronées.

Autrement dit L'extraction des motifs fréquents est une étape intermédiaire, car le résultat de cette technique est directement utilisé comme une entrée pour d'autre méthodes de découverte de connaissances.

La question qui se pose est, Quels sont les algorithmes les plus utilisés dans le cadre de "L'extraction des motifs fréquents " ?

2. Définitions et généralités

2.1. Extraction des itemsets

La recherche des régularités dans les bases de données est l'idée principale du Data Mining. Ces régularités s'expriment sous différentes formes. Dans l'analyse du panier d'achats des consommateurs, l'extraction des itemsets consiste à mettre en évidence les conccurrences entres les produits achetés c'est-à-dire déterminer les produits (les items) qui sont « souvent » achetés simultanément. On parle alors d'itemsets fréquents. Par exemple, en analysant les tickets de caisse d'un supermarché, on pourrait produire des itemsets (un ensemble d'items) du type « le pain et le lait sont présents dans X% des caddies » **[09] [10] [11]** .Le fichier comportant 5 observations (transactions) et 4 items (voir Tableau 2.1).

Tableau 2.1 : base de transactions.[09]

	Article A	**Article B**	**Article C**	**Article D**
Client 1	X	X		
Client 2	X		X	
Client 3		X		
Client 4	X		X	X
Client 5		X		

Le tableau 2.1 illustre l'exemple de paniers d'achats de certains clients.

2.2.Définition des itemsets

Dans ce qui va suivre nous allons voir nous allons voir les differents types d'itemsets selon **[09]**

- **Item** : Un item correspond à un produit. Nous avons 4 items (Article A, Article B, Article C et Article D) dans notre fichier.
- **Support** : Le support d'un item est égal au nombre de transactions dans lesquelles il apparaît.
- **Itemset** : Un itemset est un ensemble d'items. Le support d'un itemset comptabilise le nombre de transactions dans lesquelles les items apparaissent simultanément. Un itemset peut être composé d'un singleton.
- **Itemset fréquent** : Un itemset est dit fréquent si son support est supérieur à un seuil défini à l'avance, paramètre de l'algorithme de recherche.
- **Superset :** Un superset est un itemset défini par rapport à un autre itemset.
- **Itemset fermé (closed itemset)** : Un itemset fréquent est dit fermé si aucun de ses supersets n'a de support identique. Autrement dit, tous ses supersets ont un support strictement plus faible.
- **Itemset maximal (maximal itemset).** Un itemset est dit maximal si aucun de ses supersets n'est fréquent.
- **Itemset générateur (generator itemset)** : Un itemset A est dit générateur s'il n'existe aucun itemset B tel que B $\subset$A et que SUP(B) = SUP(A). Autrement dit, l'itemset est générateur si tous ses sous itemsets ont un support strictement supérieur.

3. Les algorithmes d'extraction des itemsets fréquents

3.1.L'algorithme Apriori

La plupart des algorithmes conçus pour trouver les modèles fréquents (ou les itemsets) des ensembles de données conventionnels de transaction sont basées sur l'algorithme Apriori.

L'algorithme Apriori est un algorithme d'exploration de données conçu en 1994, par Rakesh Agrawal et Ramakrishna Sikrant **[11]**

Apriori détermine les règles d'association présentes dans un jeu de données, pour un seuil de support et un seuil de confiance fixés. Ces deux valeurs peuvent être fixées arbitrairement par l'utilisateur.

Une règle d'association est une règle ayant la forme

$$ai = vi\,, aj = vj, \dots am = vm \quad \textbf{(20)}$$

ce qui s'interprète par « si les attributs $ai, aj, \dots am$ ont une certaine valeur, alors l'attribut ai prend généralement une certaine valeur vi , $a\beta$ une certaine valeur $v\beta$,… ».

La difficulté consiste notamment à trouver des règles qui sont significatives et non seulement le résultat du hasard.

Dans un jeu de données, on dispose de *xi* éléments (ex : lignes dans une table de données) connus aussi sous le nom de Transactions. Chaque élément est décrit par un ensemble d'attributs *aj* (ex : colonnes dans une ligne de données). Un attribut correspond aussi à un item. Un ensemble d'items est dit fréquent s'il correspond à un motif fréquent dans la base de transactions.

Chaque règle d'association a deux mesures : le « ***support*** » et la « ***confiance*** ».

- Le « ***support*** » d'un ensemble d'items est défini comme la fréquence d'apparition simultanée des items figurant dans l'ensemble des données.
- La « ***confiance*** » d'une règle « si condition alors conclusion » est le rapport :

$$\frac{\textbf{\textit{Nbre de données où les items de la condition et de la conclusion apparaissent simultanément}}}{\textbf{\textit{Nombre de données où les items de la condition apparaissent simultanément}}}$$

(21)

3.1.1. Exemple explicative de l'algorithme

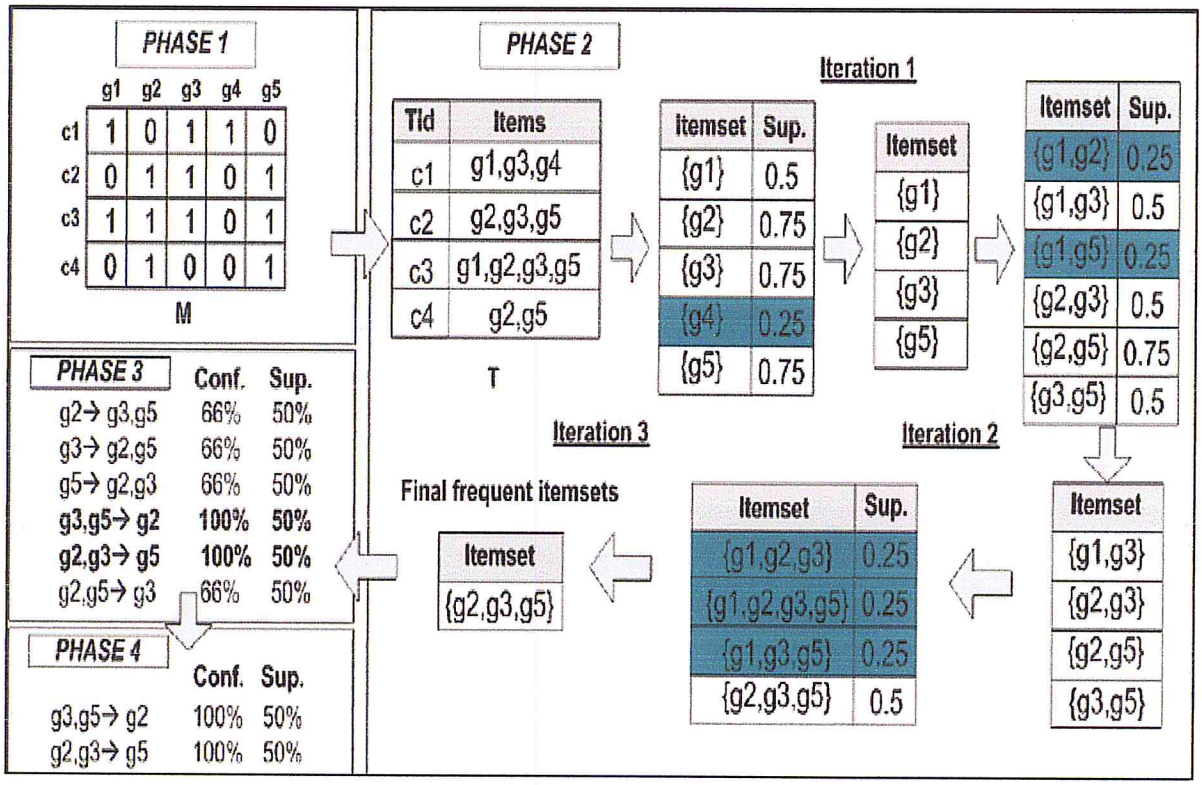


Figure 2.1 : Exemple explicatif de l'algorithme Apriori.**[12]**

3.1.1. Récapitulatif de l'algorithme Apriori [12]

Algorithme : Apriori

Calculer L1
K = 2
TANTQUE $L_{k-1} <> \varphi$ FAIRE
  C_k = apriori – gen(L_{k-1})
    TANTQUE t ∈ D FAIRE
      C_t = sousensemble (C_k , t)
        TANTQUE c ∈ C_t FAIRE
            c.count ++
        FIN TANTQUE
    FIN TANTQUE
  L_k + {c ∈ C_k | c.count ≥ minSup}
  K = k+1
FIN TANTQUE
RETOURNER $\cup_k L_k$

3.2. L'algorithme FP-Growth

Selon k.Tannir **[13]** Est un algorithme fréquent principal d'exploitation d'itemset, qui est basé sur le paradigme de croissance de modèle.

L'algorithme **FP-growth** (Fréquent Pattern growth) consiste d'abord à compresser la base de données en une structure compacte appelée **FP-Tree** (*Fréquent Pattern tree*), puis à la diviser ainsi compressée en sous projections pour représenter la base de données. Chacune de ces projections est associée à un item fréquent. L'extraction des itemsets fréquents se fera sur chacune des projections séparément.

L'algorithme **FP-growth** apporte ainsi une solution au problème de la fouille de motifs fréquents dans une grande base de données transactionnelle. En stockant l'ensemble des éléments fréquents de la base de transactions dans une structure compacte, on supprime la nécessité de devoir scanner de façon répétée la base de transactions. De plus, en triant les éléments dans la structure compacte, on accélère la recherche des motifs. **[14]**

3.2.1. Structure d'un FP-tree

Deux éléments essentiels constituent la structure d'un FP-tree qui sont :

1. Une structure sous forme d'un arbre avec une racine étiquetée ***nulle***.
2. Un index (une table des pointeurs des items fréquents).

L'arbre quant à lui est composé, d'une racine **nulle** et d'un ensemble de nœuds préfixé par l'élément représenté.

Un nœud de l'arbre est composé par :

- Le nom de l'item (nom-item). Il s'agit de l'item que représente le nœud.
- Le nombre d'occurrence (count) de transaction où figure la portion de chemin jusqu'au nœud.
- Un lien vers le nœud suivant dans l'arbre .Il s'agit d'un lien inter-nœud vers les autres occurrences du même élément (ayant le même nom-item) figurant dans d'autres séquences de transactions. Cette valeur prend la valeur ***nulle*** s'il n'y a pas un tel nœud.

L'Index est une table d'en-tête qui contient la liste des items fréquents et qui pointe sur la première occurrence de chaque élément. **[13]**

Chaque entrée dans cette table contient :

- Le nom de l'élément (nom-item).

- Le pointeur tête de la séquence des nœuds ayant ce même nom-item.

3.2.2. Construction d'un FP-Tree

La construction du FP-tree nécessite deux parcours de la base des transactions (T) et se fait de la manière suivante :

On effectue un premier parcours de la base T pour déterminer les items fréquents en fonction du support minimum fourni. Ces items seront triés par la suite par ordre décroissant de support dans une liste (L). Les items ainsi triés seront traités dans cet ordre.

Un second parcours de T est alors effectué. Chaque transaction est alors triée selon l'ordre des items dans L. Le nœud racine de l'arbre nulle est d'abord crée. Durant ce même parcours, une branche sera crée pour chaque transaction, mais des transactions ayant un même préfixe partageront le même début d'une branche de l'arbre, ainsi deux transactions identiques seront représentées par une seule et même branche.

La raison pour laquelle les items sont traités du plus fréquent au moins fréquent est que les items fréquents seront proches de la racine et seront mieux partagés par les transactions. Ceci fait du FP-tree une bonne structure compacte pour représenter les bases transactionnelles. **[13]**

3.2.3. Récapitulatif de l'algorithme Fp-growth

Algorithme : FP-growth

1 Balayer la base de transaction T une première fois

- Créer L, la liste des items fréquents avec leur support.
- Trier L en ordre décroissant du support.

2 Créer l'arbre N contenant une racine étiquetée «Null ».

3 procedure FP-growth(Fp-tree,null)

A. Si FP-tree contient un seul chemin P alors
   - Pour chaque combinaison β de P faire
   - Générer l'itemset β U α de support = minimum des supports des nœuds de β.

B. Sinon pour chaque a_i dans l'index de FP-tree faire
   - Générer l'itemset β = a_i U α de support = a_i.support.
   - Construire l'itemset condition de base de β.
   - Construire FP-tree$_\beta$
   - Si FP-tree$_\beta \neq 0$
     - FP-growth(FP-tree$_\beta$, β)

3.2.4. Etapes de la construction du FP-tree

La construction d'une structure FP-tree passe par 6 étapes principales. Les étapes 1 à 5 préparent la structure et insèrent les éléments qui doivent s'y trouver. La sixième étape consiste à valider les informations insérées dans les étapes précédentes :

✓ ***Etape 01 :*** Calculer le support minimum : Considérons la base de transactions suivante. Supposons que le support minimum est défini à 50% .

$$\textbf{\textit{Support minimum}} = (\textit{Pourcentage minimum demandé} * \textit{nombre total de transactions de la base}) = \textit{Résultat} \quad (22)$$

Si le résultat obtenu n'est pas entier, il sera arrondi, Constituant le support minimum, par conséquent tous les items da la base de transactions ayant un support inférieur à **3** occurrences minimum sera ignoré. **[13]**

✓ ***Etape 02 :*** Parcours de la base des transactions pour trouver la somme totale des différentes occurrences : Dans cette étape nous allons parcourir la base de transactions afin de calculer les fréquences des éléments qui s'y trouvent.

Par la suite, une fois les différentes fréquences obtenues, seuls les éléments dont la fréquence est supérieure au support minimum défini dans l'étape 1 seront retenus, les autres seront ignorés. **[13]**

✓ ***Etape 03 :*** Définir la priorité des éléments, puis trier les items en fonction de leur priorité, Cette étape consiste à ordonner les différents éléments en fonction de leur poids. Il s'agit de les trier en fonction de leur nombre d'occurrences. Ce tri s'effectue en ordre décroissant, l'élément ayant comptabilisé le plus grand nombre d'occurrences est placé en tête et l'élément ayant comptabilisé le moins d'occurrences est placé en queue.

A la fin de cette étape, on peut considérer que tout est prêt pour commencer la construction de la structure FP-tree avec la création et l'insertion des différents nœuds.

✓ ***Etape 04 :*** Création du nœud racine : A partir du résultat obtenu lors de l'étape précédente, nous commençons la construction de la structure FP-tree. Tout d'abord l'élément 'Racine' de l'arbre est créé. Cet élément racine ne contiendra aucun élément. Il contiendra uniquement des liens vers ses éléments enfants.

✓ ***Etape 05 ,06:*** Insertion des nœuds enfants : On commence par parcourir chaque élément de la transaction .Puis pour chaque élément de la transaction on vérifie

l'existence d'un nœud correspondant, s'il n'existe pas, le nœud est créé, dans le cas contraire le nombre d'occurrence est incrémenté. Puis pour chaque élément créé on va établir un lien depuis la table des entêtes vers l'élément inséré dans l'arbre.

A la fin du traitement de toutes les transactions de la base la structure finale de Fp-tree est illustrée par la figure suivante :

3.2.5. Exemple

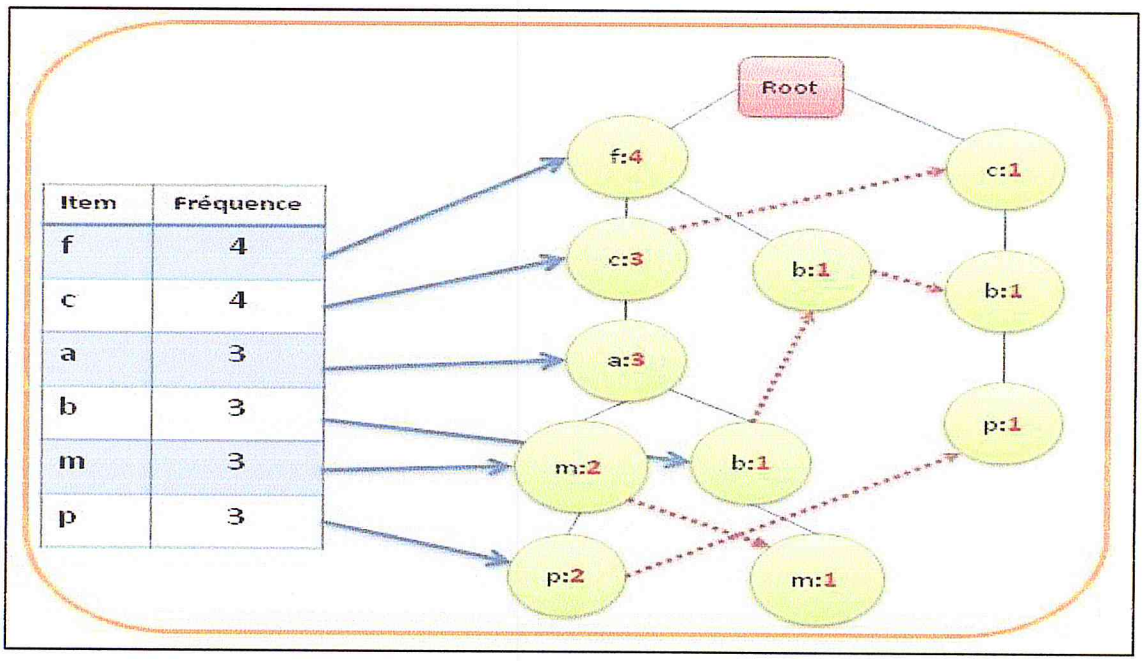


Figure 2.2 : Exemple d'état final de la structure FP-tree. **[13]**

3.3.L'algorithme Eclat

Eclat étudie des ensembles de transactions par classe d'équivalence d'items. On parle ici de placements respectivement horizontal et vertical de la base.

Cet algorithme repose sur le découpage de la base en classes d'équivalences et distribution de la charge de travail sur tous les processeurs. On considère que deux itemsets (ensemble d'items) sont dans la même classe d'équivalence s'ils désignent par les items qu'ils contiennent dans l'ordre lexicographique, possèdant un préfixe commun. Par exemple, les itemsets MERE et MEROIR sont dans la classe d'équivalence MER. Au lieu de transmettre des supports locaux ou des portions de base de données comme dans les principaux algorithmes dérivés d'Apriori.

Cet algorithme fonctionne en transmettant les listes de transactions correspondant à chaque classe d'équivalence au processeur qui s'occupe de celle-ci. **[15]**

3.3.1. Etapes de l'algorithme

✓ ***Etape 01 :*** **Phase d'initialisation :** L'algorithme commence par scanner la base afin de construire les itemsets fréquents de *taille* 2. En effet, il est possible de générer ces ensembles avec peu de coût supplémentaire par rapport à la génération des itemsets fréquents de *taille* 1, profitant ainsi du gain obtenu en évitant de scanner la base 2 fois. Cependant, l'auteur de l'algorithme précise que si la base de données contient un grand nombre d'items, il est peut-être préférable de scanner la base deux fois afin d'éliminer les items infréquentés avant de générer les itemsets de *taille* 2. **[15]**

A ce stade, on dispose de l'ensemble des itemsets fréquents $L2$.

- On scanne la base de données.
- On construit un tableau de deux dimensions indexées par les items sur la hauteur et la largeur.

Exemple : pour une base contenant 4 items, on rencontre les itemsets AB, AC, BD *et* CD chacun respectivement dans au moins une transaction de la base.

Chaque processeur calcule le support local des itemsets puis effectue une réduction de somme des résultats des autres processeurs afin de construire les supports globaux de chaque itemset de $L2$. **[15]**

Tableau 2.2 : Exemple. **[15]**

Items	A	B	C	D
A	_	_	_	_
B	**1**	_	_	_
C	**1**	**0**	_	_
D	**0**	**1**	**1**	_

Le Tableau 2.2 explique l'exemple président.

✓ ***Etape 02 :*** **La phase de transformation :** L'algorithme commence par partitionner L2 en classes d'équivalences qui seront redistribuées sur les processeurs avec une politique d'équilibrage de charge basée sur une heuristique. On effectue alors une transformation de la base afin d'obtenir, non plus une liste d'items par transaction mais une liste de transactions par item (transformation verticale de la base.)

– Partitionnement de L_2 en classes d'équivalences.

– Calcul de la charge de travail pour chaque classe d'équivalence.

La mesure est effectuée en fonction du nombre d'éléments s de la classe d'équivalence. On considère toutes les paires à traiter par la suite. Ainsi, la charge est calculée par la valeur C_s^2 .

Par exemple, si dans la classe d'équivalence $[A]$ on trouve les itemsets $AB, AC\ et\ AD$, la charge calculée sera $C_3^2 = 3$. Il s'agit alors de répartir les tâches à effectuer sur les processeurs en fonction d'une heuristique sur les charges calculées : On assigne toutes les classes d'équivalences par charge décroissantes sur le processeur qui a la charge la plus petite au moment de l'assignation.

Chaque processeur peut effectuer cette tâche indépendamment des autres puisqu'ils disposent alors de tous les supports globaux de $L2$.

Phase de transformation verticale de la base. Chaque processeur scanne sa portion locale de la base afin de construire les listes de transactions correspondant aux classes d'équivalence de $L2$. Il faut alors transmettre respectivement les listes aux processeurs chargés de la classe d'équivalence correspondante. **[15]**

✓ ***Etape 03 :*** **La phase asynchrone :** Les processeurs effectuent concurremment la construction des itemsets de tailles croissantes par intersection des listes de transactions des éléments de chaque classe d'équivalence entre eux. Eliminant les itemsets de support insuffisant, on réduit rapidement le travail à effectuer en même temps qu'on augmente la taille des itemsets construits. C'est du moins ce qu'il se passe sur des données réelles.

✓ ***Etape 04 :*** **La phase de réduction finale :** La dernière tâche de l'algorithme consiste en l'accumulation et la réunion des résultats de chaque processeur.

3.3.2. Récapitulatif de l'algorithme Eclat [15]

Algorithme : Eclat
Phase d'initialisation o Scanne la partition locale de la base. o Calcul des comptes locaux des itemsets de taille 2. o Construction des comptes globaux de L_2 **Phase de transformation** o Partitionnement de L_2 en classes d'équivalences. o Distribution de L_2 sur les processeurs par classe d'équivalence. o Transformation de la base locale en base verticale. o Envoie des listes de transactions aux autres processeurs. o L_2 **local** = listes des transactions des autres processeurs. **Phase asynchrone** o **Pour** chaque classe d'équivalence E_2 dans L_2 **local** *Construction*(E_2) **Phase final de réduction** o Regroupement des résultats et calcule des associations

4. Conclusion

Dans ce chapitre nous avons vu les algorithmes d'extractions d'itemsets fréquents à partir de données parfaites, ou on a répondu sur la question posée dans l'introduction de ce chapitre, nous avons aussi expliqué leur fonctionnement. L'efficacité de ces algorithmes d'extraction n'est plus un obstacle pour des données parfaites, mais encore il est nécessaire de développer des méthodes quand il s'agit de données imparfaites. C'est ce que nous allons développer dans les prochains chapitres.

Chapitre 3 :

«L'extraction des itemset fréquents à partir de données imparfaites »

1. Introduction

L'extraction de données est une technique qui utilise une variété d'outils d'analyse des données pour découvrir, Des motifs fréquents et des relations cachés c'est pourquoi, nous aborderons dans ce chapitre une recherche bibliographique et nous exposerons une synthèse des travaux qui traitent le problème d'extraction des itemsets fréquents à partir de données imparfaites.

2. Extraction des motifs fréquents à partir de données imparfaites

Les données imparfaites sont représentées par trois grands types d'imperfection comme vu dans le premier chapitre (incertitude, imprécision, incomplétude) **[01]**. Il existée dans la littérature différentes méthodes qui traitent l'extraction de motifs fréquents à partir des données imparfaites, cependant chaque aspect de l'imperfection est traité d'une façon indépendante.

2.1.Extraction des itemsets fréquents à partir de données incertaines

Il existe plusieurs méthodes pour calculer les itemsets fréquents à partir de données incertaines dans ce qui va suivre nous allons voir ces mesures de calcules.

2.1.1. Mesures de calcule des fréquences des itemsets incertains

Une valeur est dite incertaine si la validité de la donnée est mise en question pour la validation de l'information. Ceci reste une étape transitoire pour la validation de la donnée réservé à des avancements intermédiaires de la validation. **[16]** C'est pour cela les algorithmes d'extractions des itemsets fréquents à partir des données incertaines utilisent trois mesures pour calculer les itemsets fréquents illustre dans la figure 3.1 :

- Expected Support.
- Probabilistic Support.
- Evidential Support.

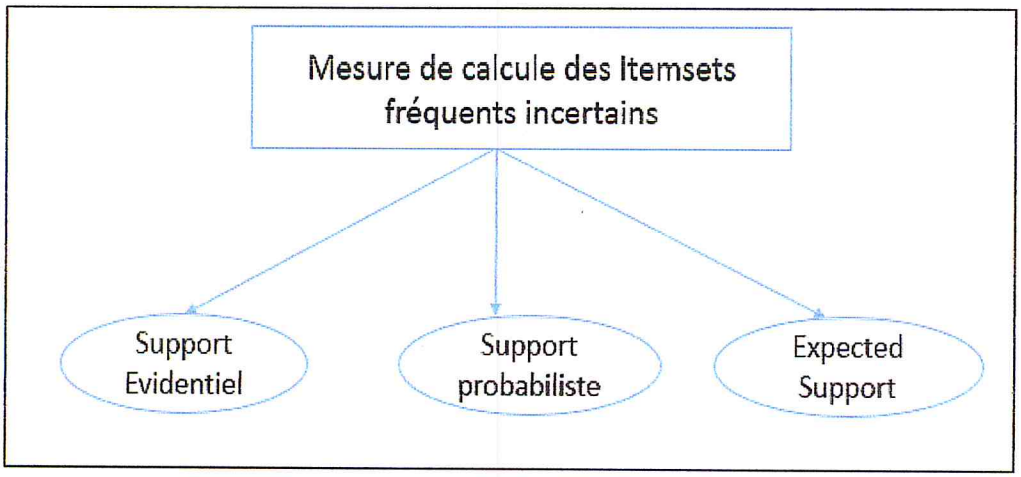


Figure 3.1 : Les mesures de calcule des itemsets fréquents. [16]

2.1.1.1. Expected support

Expected support d'un itemests X dans la base de données incertaine est la somme de produit de probabilité P (X, t j) de X dans la transaction t j sur tous les n transactions dans la base de données. [16]

Donc l'expression de la fonction Expected Support est présentée comme suit :

$$expSup(X) = \sum_{j=1}^{n} P(X, tj) = \sum_{j=1}^{n} \left[\prod_{x \in X} P(x, tj) \right] \qquad (23)$$

Lorsque des éléments x ∈ X dans chaque transaction Tj sont indépendants.

Remarque : l'itemset X est un fréquent si son Expected support supérieur ou égale le seuil minsup.

$$\textit{ExpSup (X)>= minsup.} \qquad (24)$$

2.1.1.2. Probabilistic Support

Dans les bases de données de transactions incertaines, le support d'un ou plusieurs éléments ne peut pas être représenté par une valeur unique, mais plutôt, doit être représenté par une distribution de probabilité discrète. [16]

Soit T est la base de données de transaction et W est l'ensemble des éléments possibles de T, le $P_i(X)$ est la probabilité de support d'un itemset X tel que X a le soutien i.

$$P(X) = \sum_{w,W,(S(X,wj)=i)} P(wj) \qquad (25)$$

Où S (X, wj) est le support de X dans un élément wj.

Le support probabiliste d'un itemset X dans une base de données de transaction incertaine T est donné par les probabilités de support de X (P_i (X)) pour toutes les valeurs possibles de seuil i ∈{0,..... |T|}.

$$\sum_{0\leq i\leq |T|} Pi(X) = 1.0. \qquad (26)$$

Soit I l'ensemble de tous éléments possibles et T représente la base de données incertaines, où une transaction tj ∈ T est un ensemble des éléments incertains, à savoir, tj ⊆ I. Contrairement à la base de données précise traditionnelle, chaque item xi ∈ tj est associé à une probabilité existentielle P (xi, tj) ∈ [0,1], ce qui dénote la probabilité que xi est réellement présent dans tj. Pour un itemset X ⊆ tj, basé sur l'hypothèse commune que les éléments de X sont indépendants la probabilité existentielle

$$P (X \subseteq tj) = \prod x\in XP (x, tj). \quad (27)$$

Le seuil attendu (Expected support) de X est alors la somme de la probabilité existentielle sur toutes les transactions. **[17].[18].**

$$expSup (X) = \sum tj \in TP (X \subseteq tj). \qquad (28)$$

La probabilité fréquente P (X) de X peut être calculée par sommer Pi (X) pour tout i ≥ minsup :

$$P(X) = \sum_{i=minsup}^{|T|} Pi(X) \qquad (29)$$

Où Pi (X) enregistre la probabilité de X qui se produit exactement dans une transaction :

$$Pi(X) = \sum_{\substack{S\subseteq T \\ |S|=i}} \left(\prod_{t\in S} P(X \subseteq t) \prod_{t\in T-S} (1 - P(X \subseteq t)) \right) \qquad (30)$$

Si P (X) ≥ minprob, alors X est un itemset fréquent probabiliste.

On donne :

(i) une base de données de transaction incertaine T.

(ii) un seuil minimum de support minsup.

(iii) un seuil minimum de la probabilité fréquente minprob, le problème d'extraction de motifs fréquents probabilistes est de trouver tous et seuls les motifs fréquents probabilistes ; à savoir, trouver tous les X tel que P (X) ≥ minprob.

Remarque : Un itemset X est un fréquent probabiliste si son existence dans une transaction *minsup* est supérieure ou égale au seuil d'utilisateur spécifique *minprob*.**[20]**

$$P(sup(X) >= minsup) >= minprob \quad (31)$$

2.1.1.3. Evidential support

Hewawasam et al **[19],** ont introduit une nouvelle approche pour support itemset informatique et appliqués sur **un Fréquent Itemset Maintenance (*FIM*)** problème. Toutes les méthodes **[19][21]** ont été basés sur le produit cartésien entre ***BBAs.*** Le support d'un itemset $\boldsymbol{X = \prod i \in [1 \ldots . n]}$ tel que xi est un item évidentiel appartenant au cadre de discernement $\boldsymbol{\theta i}$. Étant donné que les items ne partagent pas le même cadre de discernement, toute règle de fusion ne peut pas être appliquée. Dans ce qui suit, nous étudions le support de croyance introduit par Dalvi et al **[18]** calculé par l'équation suivante :

$$mj(X) = \prod_{x_i \in X} mij(x_i) \quad (32)$$

Où $\boldsymbol{mj\ (X)}$est le produit cartésien de tous ***BBA*** dans la transaction $\boldsymbol{Tj}$. Ainsi, le ***BBA*** de l'itemset $\boldsymbol{X}$ exprimé dans l'ensemble ***EDB*** devient :

$$m_{EDB}(X) = \frac{1}{d} \sum_{j=1}^{d} mj(X) \quad (33)$$

Ensuite, le support de $\boldsymbol{X}$ dans la base de données ***EDB***devient :

$$Support_{EDB}(X) = Bel_{EDB}(X) \quad (34)$$

Le produit cartésien d'un support à base, présenté ci-dessus, remplit plusieurs propriétés telle que la propriété anti-monotonie. Une mesure de support satisfaisant la propriété anti- monotonie consiste dans le fait qu'un itemset qui contient un itemset fréquent est également fréquent. L'inverse est vrai, tous les itemsets constituant un des plus fréquents sont également fréquents. Avec cette propriété satisfaite, la construction d'un algorithme Apriori devient simple **[20].**

2.1.2. Les algorithmes d'extraction des itemsets fréquents

Il existe plusieurs algorithmes qui utilisent les mesures de calculs de fréquences vu précédemment dans le chapitre 02 pour l'extraction des itemsets fréquents à partir de données parfaite.

Nous avons recensées ces algorithmes avec les mesures de calcule vu précédemment Schéma 3.1) pour faire l'extraction des itemsets fréquents à partir de données incertaines comme vu dans la figure 3.2.

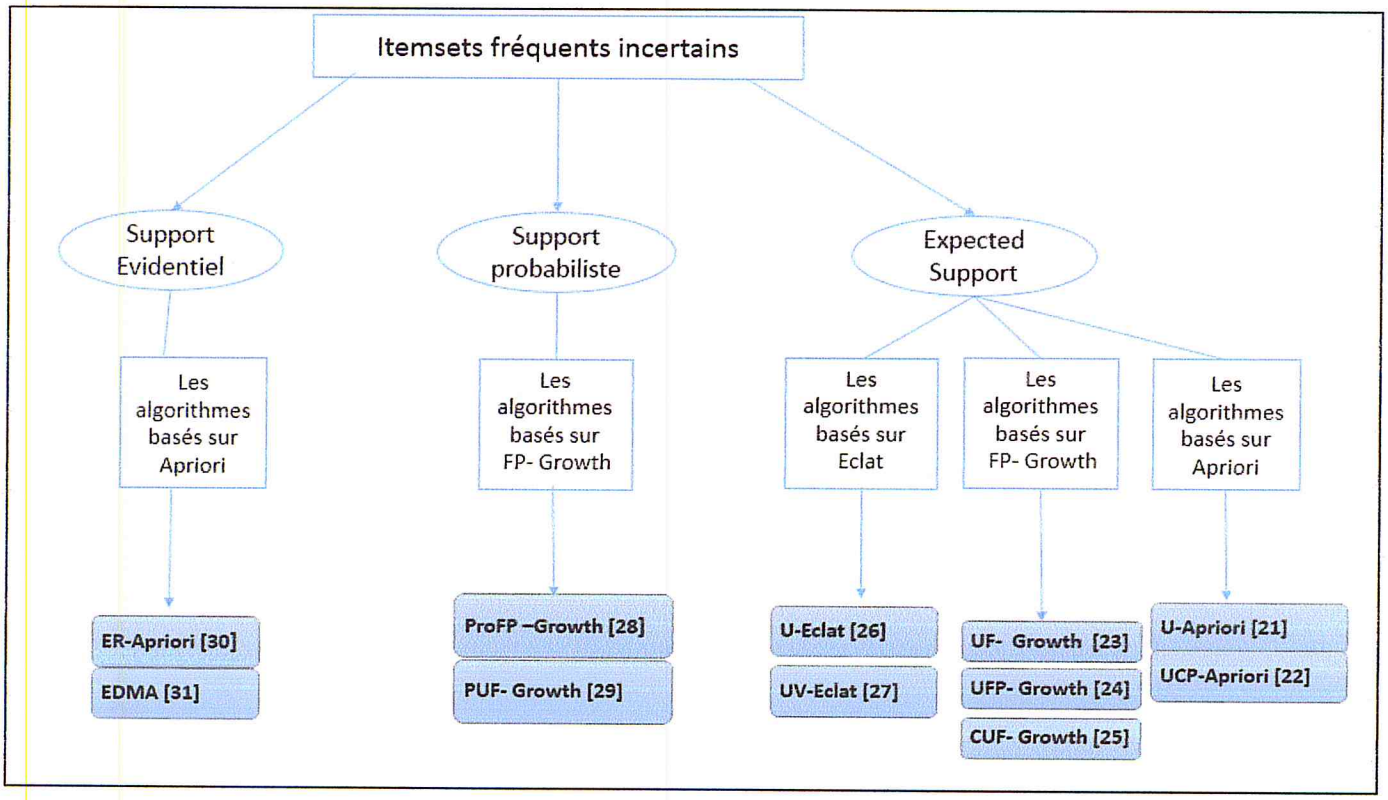


Figure 3.2 : Classification des algorithmes selon le support utilisé.

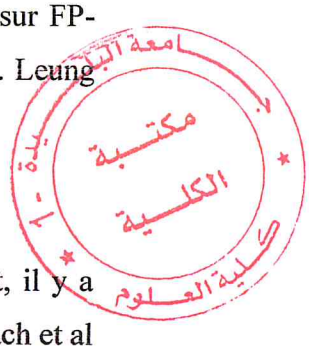
2.1.1.1. Approche basé sur l'Expected Support et Probabiliste Support

De multiples algorithmes utilisent les mesures de calculs des itemsets fréquents a partir de données incertaines mais d'un algorithme a un autre ces mesures changent dans le but d'extraire la fréquence comme vu dans le précédant chapitre ou nous avons vu trois grand algorithmes (Apriori, FP-Growth et Eclat) ces derniers utilisent les mesures de calcules des itemsets fréquents incertains comme suit :

- Selon M.A, BachTobji et al **[21]** et Nguyen **[22]** les algorithmes U-Apriori et UCP-Apriori sont basé sur l'algorithme Apriori vu dans le chapitre 2 utilisent l'Expected Support pour l'EIF à partir de données incertaines.
- Selon C. Leung et al **[23]** UF-Growth et C. C. Aggarwal **[24]** UFP-Growth et C. Leung et al **[25]** CUF-Growth sont des algorithmes basés sur FP-Growth qui utilisent l'Expected Support pour l'EIF à partir de données incertaines.

- Selon T. Calders et al **[26]** U-Eclat et B. Budhia **[27]** UV-Eclat basés sur Eclat utilisent aussi l'Expected Support pour l'EIF à partir de données incertaines.

Nous avons aussi Probabiliste Support qui utilise des algorithmes basés sur FP-Growth tel que Pro-FPGowth, selon T. Bernecker **[28]** et PUF-Growth selon C. Leung et al **[29]** pour l'EIF des données incertaines.



2.1.1.2.Approche basé sur le support évidentiel

L'extraction de données évidentielles n'a pas eu tant d'attention. En effet, il y a peu d'ouvrages qui ont abordé le thème de l'extraction des itemsets fréquents. Bach et al **[33]** ont proposé d'extraire des itemsets fréquents basés sur la représentation verticale de la base de données évidentielles en utilisant une structure de données de listes d'identificateurs d'enregistrement (RidLists). Cette structure de données contribue à accélérer le comptage de support évidentiel d'itemsets. La fonction de croyance basée sur le support évidentiel a été utilisée dans ce travail. Les mêmes auteurs ont proposé dans **[32]** une méthode basée sur la structure d'arbre pour extraire des itemsets fréquents à partir de bases de données évidentielles qui incluent exactement un attribut évidentiel. Ils ont utilisé (BIT) Belief itemset tree qui est une structure de données pour stocker une base de données évidentielles de manière compressée. Des itemsets fréquents sont générés par la suite à partir de cette structure de données arborescente.

L'algorithme de EDMA proposé par Samet *et al* **[31]** est un Apriori-based algorithme qui extrait les itemsets fréquents à partir des bases de données évidentielles en utilisant évidential precise support pour le calcul de la fréquence des itemsets.

Samet et al **[30]** ont étendu le concept de base de données évidentielles à des bases de données fiables qui tiennent en compte la fiabilité des données. Par conséquent. Pour les itemsets fréquents de ce type de données les auteurs ont proposé une nouvelle définition pour le support évidential des données fiables. La version classique de l'algorithme *Apriori* a été mise à jour pour extraire les itemsets fréquents à partir des données fiables. La nouvelle version est appelée Evidential Reliable Apriori (ER-Apriori).

2.2. Extraction des itemsets fréquents à partir de données imprécises

Il est généralement reconnu que les diverses relations du monde réel sont fondamentalement floues. Par conséquent, le concept d'ensembles flous **[34]** peut gérer l'incertitude et l'imprécision des données, car il peut être utilisé pour extraire des

données incertaines. Cependant, ce concept a été largement utilisé pour les règles d'associations à partir de données imprécises plutôt que des données incertaines.

L'extraction des motifs fréquents est la première étape de l'extraction des règles d'association pour cela nous avons trouvé comme mesure de calcul pour ces itemsets fréquents le Fuzzy Support que nous allons voir ci-dessous.

2.2.1. Fuzzy Support

On dénote fuzzy base de donnée par le triple *FBD* = {A, O, R} qui fuzzy base de données ou **R** dénote le rapport floue entre un objet et une transaction exprimé par une fonction d'adhésion $\mu Tj\ (i)$ tel que **[35]**

$$\mu Tj\ (i) = \propto\ (\propto \in [0,1]) \quad \textbf{(35)}$$

Ou la fonction $\mu Tj\ (i)$ évalue le degré d'adhésion de l'objet considéré à la transaction Tj.

Le calcul de Support dans de telles bases de données est fait en employant la fonction de *count (i) de la* façon suivante :

$$count(i) = \sum_{j=1}^{d} \mu Tj(i). \quad \textbf{(36)}$$

Le support d'un item i dans la base de donnée floue est comme suite

$$support(i) = \frac{count(i)}{d} \quad \textbf{(37)}$$

Ainsi, pour un itemset *X* de la taille *q* tels que le $xi \in X$ et $i \in [1, q]$ le Support devient :

$$support(X) = \frac{\sum_{j=1}^{d} \min\{\mu Tj(xi), i=1...q\}}{d} \quad \textbf{(38)}$$

2.3. Extraction d'itemsets fréquents à partir de données incomplètes

Une donnée est dite incomplète si elle contient au moins une valeur manquante pour cela il existe différentes méthodes pour traiter l'incomplétude tel que l'imputation de données ou on remplace les données manquantes par les données probables et/ ou possibles par la suite les données deviennent incertaines. C'est-à-dire quand on élimine le problème d'incomplétude un autre problème apparait celui de l'incertitude alors on applique les méthodes connus pour les itemsets incertains comme vu précédemment.

3. Conclusion

Dans ce chapitre nous avons présenté plusieurs supports pour les différents types de données imparfaites ainsi que les déférents algorithmes qui traitent ces imperfections, notre perspective dans le prochain chapitre est de trouver une seule mesure pour le calcul des itemsets fréquents à partir de données imparfaites.

Chapitre 4 :

« Approche proposée »

1. Introduction

Dans ce chapitre, nous proposons un algorithme inspiré de l'algorithme Apriori, avec nos propres contributions qui sont la proposition d'une structure efficace, pour pouvoir manipulé le contenus des transactions, cette algorithme est nommé Rough-DS-Apriori comme son nom l'indique il est basé sur les concepts de deux grandes théories la théorie de évidence et la théorie des ensemble d'approximation ce qui va nous permettre l'extraction des itemsets fréquents à partir de données imparfaites.

2. Architecture du système

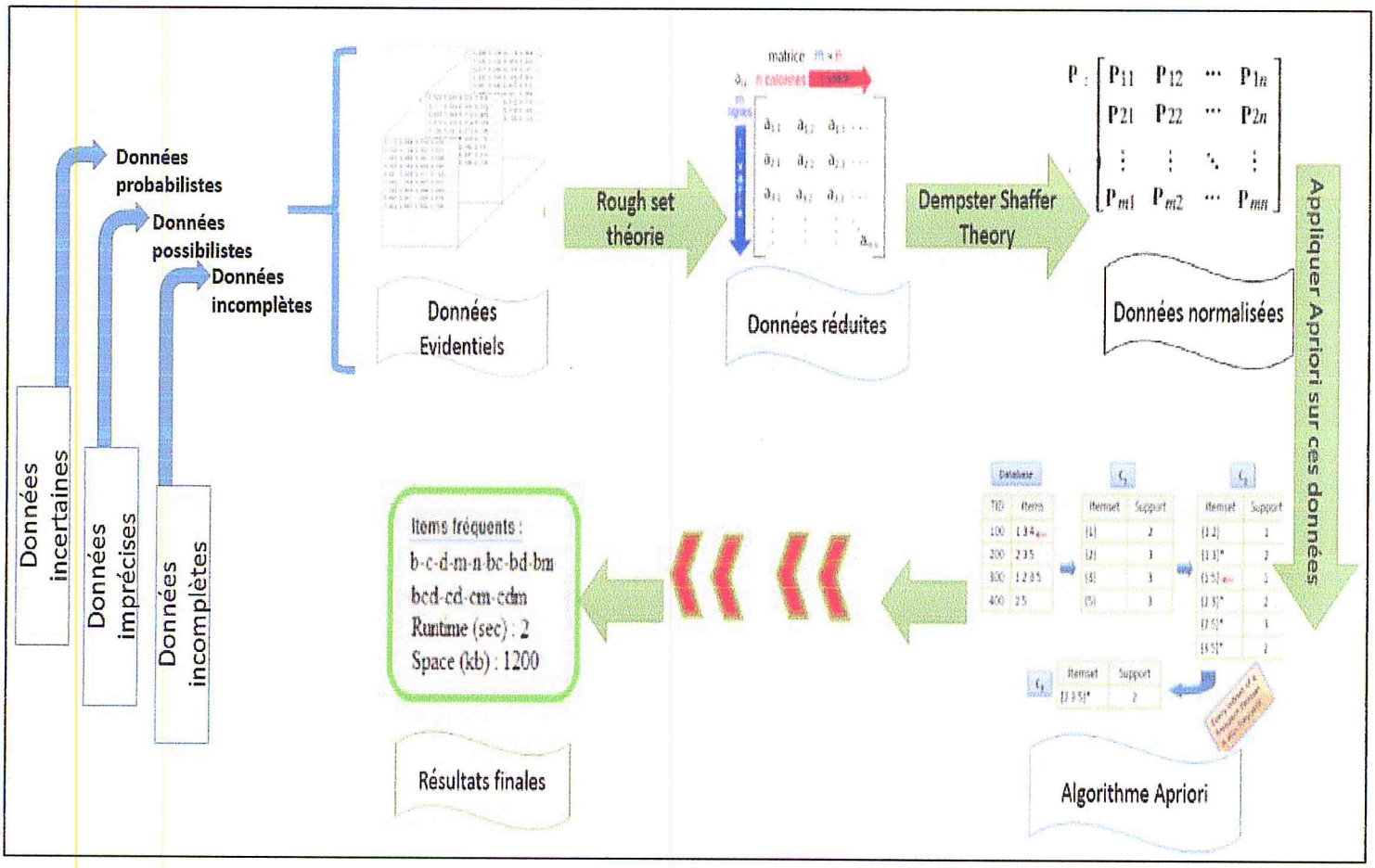


Figure 4.1 : Schéma global.

La figure 4.1 représente notre architecture du système.

2.1. Scenario

Initialement les données imparfaites sont représentées par des données évidentielles nous avons utilisé rough set théorie pour réduire ces données tout en gardons leurs sens puis nous avons utilisé le support Evidential de Dempster Scheffer

pour la normalisation de ces derniers et en fin pour extraire les itemsets fréquents de ces données on utilisera apriori.

3. Pseudo algorithme

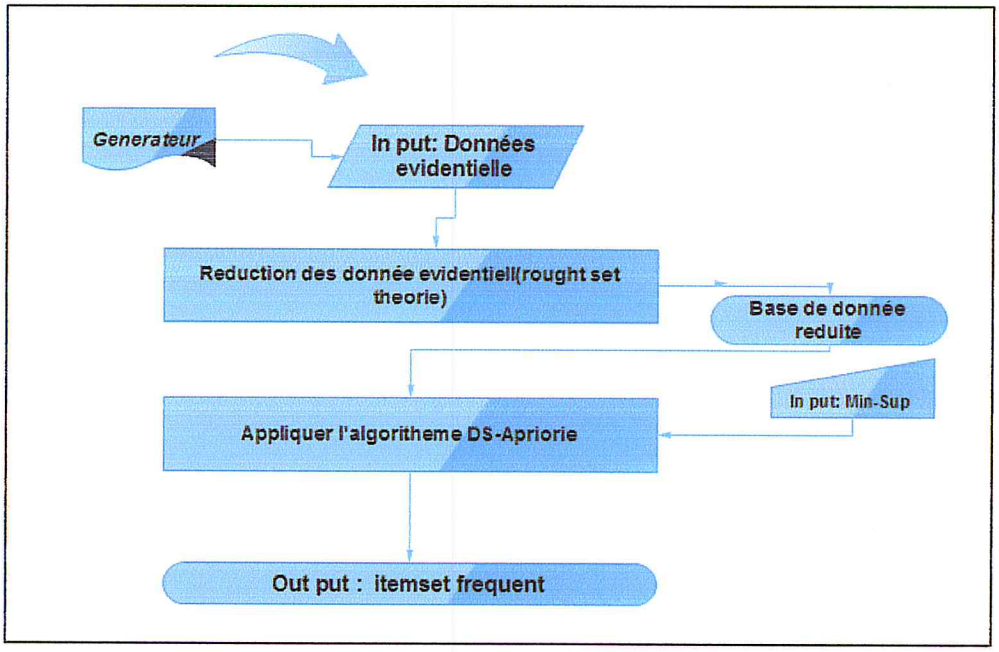


Figure 4.2: Rough-DS-Apriori.

3.1. Scenario

Entrées **:** Donnée évidentielles, minSup

Sorties : Itemsets fréquents

Etapes :

Etape 01 : consiste a importé les données évidentielles générer et stocker dans un fichier.

Etape 02 : est basée sur la théorie des ensembles approximative, ou on doit réduire notre base de données évidentielles.

Etape 03 : on applique l'algorithme Rough-DS-Apriori sur la base de donnée évidentielles réduite (c.-à-d. le résultat de la deuxième étape) apriori prend en entrée une valeur minSup représentant un seuil de fréquence.

Récapitulatif de l'algorithme

Algorithme : Rough-DS-Apriori

- **Phase de génération des données :**

Générer des données évidentielles.

- **Phase de réduction des données évidentielles :**

Entrées : *données évidentielles*

Pour chaque Entités (transactions) *Tj* **faire**

Pour chaque attribut (objet) *Ai* **faire**

Calculer la somme d'ignorance *I*.

Calculer la somme d'existence *P*.

Fin Pour.

Si $P < I$ **alors** éliminer la transaction.

Fin Si.

Fin Pour.

Sortie : données réduites.

- **Phase d'extraction des itemsets fréquents :**

Entrées : *données réduites, minSup.*

Pour chaque **item** calculer Evidential precise support.

S ← Evidential precise support.

Si $(S \geq minSup)$ **alors** *itemset fréquent.*

Fin Si

Fin Pour.

4. Données évidentielles

Selon Agrawal et al **[36]** une base de données évidentielles permet le stockage de données incertaines et imprécises et même incomplètes qui sont modélisées à l'aide de la théorie de Dempster-Shafer. Une base de données évidentielles notée **BDEs** contient **n** attributs et **d** lignes. Chaque attribut i $(1 \leq i \leq n)$ a un domaine Θi de valeurs discrètes. La valeur de la colonne i pour la ligne j est dite valeur évidentielle et notée **Vij** .

4.1. Définition (valeur évidentielles)

Soit *Vij* une valeur évidentielle qui correspond à la colonne *i* et à la ligne *j*. *Vij* correspondant à un corps d'évidence composé du cadre de discernement Θi (le domaine de la colonne *i*), de l'ensemble *Fij* des éléments focaux et de la fonction de masse *mij* qui est définie comme suit :

$$mij : 2^{\Theta i} \rightarrow [0, 1] \text{ avec :}$$
$$mij(\emptyset) = 0 \text{ et } \sum mij(x) = 1 \text{ avec } x \subseteq \Theta i \quad (39)$$

Tableau 4.1 : Exemple d'une base de données évidentielle.

	A	B
T01	A1{0.5}, A2{0.5}, Θ{0.0}	B1{1}
T02	A1{0.2}, (A1,A3){0.1}, Θ{0.7}	B2{0.1} , Θ{0.9}
T03	A1{0.1}, (A2,A3) {0.4} , Θ{0.5}	B3{0.5} , Θ{0.5}
T04	A1{0.8} , Θ{0.2}	B4 {1}

Le tableau 4.1 illustre un exemple Pour deux objets A et B et quatre transactions, qui présentent les différents types d'imperfections. Selon Bach tobji et al **[36]** les données imparfaites peuvent être modélisées par des données évidentielles comme suit :

- ***Données parfaites*** : Dans une base de données évidentielles, la valeur d'un attribut est un corps d'évidence comme nous l'avons précisé auparavant. Quand il inclut exactement un élément focal qui est singleton, et dont la masse est par conséquent égale à l'unité, nous parlons de donnée parfaite. Le tableau 4.1, exemple la valeur de la colonne *B* est parfaite dans les transactions 1 et 4.

- ***Données probabilistes*** : Quand le corps d'évidence inclut des éléments focaux singletons, là nous parlons d'une valeur probabiliste. Dans le tableau 4.1 la valeur de la colonne *A* dans la première transaction et la dernière sont probabilistes même chose pour la valeur de l'attribut B dans la deuxième et troisième transactions.

- ***Données possibilistes*** : Lorsque le corps d'évidence inclut des éléments focaux imbriqués, nous parlons de valeur possibiliste. En effet, la fonction π en théorie de possibilité correspond à la fonction *pl* en théorie de Dempster-Shafer. Dans ce cas, la valeur de l'attribut *A* dans la deuxième transaction est possibiliste.

- ***Données manquantes :*** Si la valeur d'un attribut *i* pour une ligne *j* est manquante, alors la ***bba ou Vij*** inclut un seul élément focal qui coïncide avec Θi, soit le domaine de l'attribut *i*, avec une masse égale à l'unité.

- ***Données évidentielles*** : A l'instar de la valeur de l'attribut *A* dans la troisième transaction qui n'est ni parfaite, ni probabiliste, ni possibiliste, ni manquante.

5. Rough-DS-Apriori

5.1. Phase de réduction des données : les principes de la théorie des ensembles d'approximation peuvent interpréter selon Nguyen dans **[37]** la prise de décisions dans un système d'information ou les données sont imparfaites comme suit :

Un système d'information : définit par la paire I= (U, A) où U est un ensemble fini non vide d'objets appelés l'univers et A est un ensemble fini non vide d'attributs tels que fa : U → V_a pour chaque a € A La valeur discrète non vide V_a régler s'appelle le domaine de a.

Une table de décision : N'importe quel système d'information prenant la forme

$$I= (U, A \cup \{d\}) \text{ et } V_d= \{d_1.....d_k\} \quad \textbf{(40)}$$

Dénote l'ensemble de valeur de l'attribut de décision.

L'espace d'approximation :

- Approximation supérieure : $\overline{B}X = \{x|[x]_B \cap X \neq \emptyset\}$ décrit les objets qui appartiennent probablement au sous ensemble d'intérêt.
- Approximation inferieure : $\underline{B}X = \{x|[x]_B \subset X\}$ décrit les objets de domaine qui sont connus avec certitude pour appartenir au sous ensemble d'intérêt.
- Région limite : $BN_B(X) = \overline{B}X - \underline{B}X$ si la région limite est non vide on peut dire qu'une décision est rough.
- Région frontière : $U - \overline{B}X$.

5.1.1. Principe de la réduction

Roughset va nous permettre la réduction de nos données évidentielles, tout en gardant leurs sens, cependant nos données ne comportent aucune décision sur laquelle on pourra s'appuyer et faire la réduction, pour cela nous avons accompagné chaque transaction par une décision basée sur la probabilité de son existence et de son ignorance comme suite :

Supposons *U* l'ensemble de nos transactions, *A* l'ensemble de nos objets, la condition de réduction est défini comme suit : si la probabilité d'ignorance domine la probabilité d'existence alors la transaction sera éliminer.

Tableau 4.2 : le système d'information.

	A	B	Décision
T01	A1{0.5},A2{0.5} ,Θ{0.0}	B1{1}, Θ{0.0}	P=2 I=0
T02	A1{0.2},(A1,A3){0.1}, Θ{0.7}	B2{0.1} , Θ{0.9}	P=0.4 I=0.16
T03	(A1){0.1}, (A2,A3) {0.4} , Θ{0.5}	B3{0.5} , Θ{0.5}	P=1 I=1
T04	A1{0.8} , Θ{0.2}	B4 {1}	P=1.8 I=0.2
T05	A1{0.2} ,A1A2{0.1} ,A3{0.1} Θ{0.6}	B1{0.3}, B2{0.2}, Θ{0.2}	P=0.9 I=1.1
T06	A1{0.3} ,Θ{0.7}	B1{0.3}, B2{0.1}, B{0.6}	P=0.7 I=1.3

Le tableau 4.2 illustre le calcul de la somme des masses **P** et la somme de l'ignorance **I** qui représentent respectivement l'existence et l'ignorance des objets par rapport à chaque transaction.

- **P** : représente la probabilité d'existence des objets *j* pour une transaction *Ti.*
- **I** : représente la probabilité d'ignorance des objets *j* pour une transaction *Ti.*

Cette phase va nous permettre la prise de décision en ce qui concerne nos transactions puis la construction de nos ensembles d'approximations, montré comme suit :

Tableau 4.3 : Une table de décision.

	A	B	décision
T01	A1{0.5},A2{0.5}, Θ{0.0}	B1{1}, Θ{0.0}	**Non**
T02	A1{0.2},(A1,A3){0.1}, Θ{0.7}	B2{0.1} , Θ{0.9}	**Oui**
T03	(A1){0.1}, (A2,A3) {0.4} , Θ{0.5}	B3{0.5} , Θ{0.5}	**Rough**
T04	A1{0.8} , Θ{0.2}	B4 {1}	**Non**
T05	A1{0.2} ,A1A2{0.1}, A3{0.1} Θ{0.6}	B1{0.3}, B2{0.2} ,Θ{0.2}	**Oui**
T06	A1{0.3}, Θ{0.7}	B1{0.3} ,B2{0.1} B{0.6}	**Oui**

Maintenant le Tableau 4.3 représente Une table de décision complet car il comporte un attribut de décision qui va nous permettre d'appliquer les principes de la théorie Rough set par la suite on réduit les données évidentielles pour cela on va

construire nos ensembles d'approximation à partir de ces données montrées comme suit :

- Approximation supérieure : $\overline{B}X$ = {T02, T03, T05, T06}.
- Approximation inferieure : $\underline{B}X$ = {T02, T05, T06}.
- region limite : $BN_B(X) = BX - \overline{\underline{B}}X$ = {T03}.

Remarque : on dit que la transaction T03 est rough car la région limite est non vide.

Tableau 4.4 : données évidentielles réduites.

	A	B
T01	A1{0.5},A2{0.5} ,Θ{0.0}	B1{1}, Θ{0.0}
T03	(A1){0.1}, (A2,A3) {0.4} , Θ{0.5}	B3{0.5} , Θ{0.5}
T04	A1{0.8} , Θ{0.2}	B4 {1}

Apres avoir appliqué la théorie des ensembles d'approximations nous avons pu réduire considérablement nos données en éliminant les données qui appartiennent à l'ensemble d'approximation inferieur sans pour autant causé une perte d'information.

5.2.Phase d'extraction des itemsets fréquents

Pour cette phase on utilisera l'algorithme apriori cependant nous avons besoin d'une mesure de calcul qui soit fiable pour cela nous avons choisi le support évidentiel nommé **precise évidential support** pour sa précision dans les calculs étant donnée la nature imparfaite de nos données.

Selon **[03]** quelque principe de la théorie d'évidence :

- $\Theta = \{\theta 1, \theta 2, \ldots, \theta n\}$ un ensemble fini non-vide incluant n hypothèses,
- *La fonction de masse* élémentaire m d'une hypothèse$X \subseteq \Theta$, notée $m(X)$[repressente la partie du degré de croyance placée sur X et qui n'a pas été distribuée aux sous-ensembles de X]. La masse de croyance élémentaire, définie par : $\boldsymbol{m}: \boldsymbol{2^{\Theta}} \rightarrow [\mathbf{0}, \mathbf{1}]$
- *éléments focaux* leur ensemble est noté $\boldsymbol{Foc}$ [Les sous-ensembles de $\boldsymbol{\Theta}$ sur lesquels sont placés des masses strictement positives]
- Le couple $\{\boldsymbol{\Theta}, \boldsymbol{Foc}, \boldsymbol{m}\}$ est appelé *corps d'évidence*

- *La fonction de croyance* **bel** est définie par $\boldsymbol{bel(A) = \sum_{B \subseteq A} m(B)}$
- *Une Base de données évidentielles (EDB) :*
  - ✓ permet le stockage de données imparfaites qui sont modélisées à l'aide de la théorie de Dempster-Shafer.
  - ✓ contient $\boldsymbol{n}$ attributs et $\boldsymbol{d}$ lignes.

La valeur de la colonne$\boldsymbol{i}$ pour la ligne $\boldsymbol{j}$ est dite **valeur évidentielle** et notée$\boldsymbol{Vij}$.

5.2.1. Evidential précise support

Dans le chapitre précédant nous avions scruté un état d'art des supports évidentiels cependant selon Samet et al **[35]** le précise support nous donne plus de précision en ce qui concerne la fiabilité des calculs par rapport aux travaux existants c'est pourquoi nous avons choisi de l'introduire dans notre travail d'extraction des itemsets fréquents surtout que la nature de nos données est imparfaite la précision dans les calculs doit être de rigueur.

Considérons une base de données évidentielles et l'itemset $X = x1 \times \cdots \times xn$ constitué par le produit des items (éléments focaux) $\boldsymbol{xi}$ $(\boldsymbol{1 \leq i \leq n})$ du cadre exclusif de discernement $\boldsymbol{\theta i}$.

Le degré de présence d'un élément **xi** dans une transaction $\boldsymbol{Tj(BBA)}$ peut être mesuré comme suit : $Pr: 2^{\theta} \rightarrow [0,1]$

$$\Pr(x_i) = \sum_{x \subseteq \theta_i} \frac{|x_i \cap x|}{|x|} \times m(x) \qquad \forall x_i \in 2^{\theta i} \tag{41}$$

Comme illustré ci-dessus, le $\boldsymbol{Pr}$ (.)Mesure permet de calculer la présence de $\boldsymbol{xi}$ en un seul La masse de croyance élémentaire note **bba** définie par : $\boldsymbol{m: 2\Theta \rightarrow [0,1]}$

Soit $\boldsymbol{\Theta = \{\theta 1, \theta 2, \ldots, \theta n\}}$ un ensemble fini non-vide incluant $\boldsymbol{n}$ hypothèses). La mesure $\boldsymbol{Pr}$ est égale à la probabilité pianistique de $\boldsymbol{x \in \theta i}$. L'évidentielle support d'un itemset calculé comme suit :

$$Support_{Tj}^{Pr}(X) = \prod_{x_i \in \theta_i, i \in [1 \ldots n]} \Pr(x_i) \tag{42}$$

$$Support_{EDB}(X) = \frac{1}{d}\sum_{j=1}^{d} Support_{Tj}^{Pr}(X) \qquad (43)$$

6. Conclusion

Dans ce chapitre, nous avons introduit deux preuves théoriques sur lesquelles nous avons basé notre approche pour le problème d'EIF à partir de données imparfaites, nous avons présenté un nouvel algorithme appelé Rough-DS-Apriori pour l'extraction des itemsets fréquents. Le chapitre suivant est consacré à démontrer l'efficacité de notre méthode grâce au résultat obtenu.

Chapitre 5 :

« Test et Validation»

1. Introduction

Après avoir décrit notre solution, nous aborderons dans ce chapitre la partie test et validation de notre application. En première partie, nous présenterons l'environnement de travail et les outils de développement utilisés. Ensuite, nous présentons notre jeu d'essai. Enfin, nous présenterons notre application ainsi que son fonctionnement et les résultats.

2. Environnement de développement

2.1. L'environnement matériel

Pour développer notre plateforme, on a utilisé une machine, configurée comme suit :

Processeur : Intel (R) Core (TM) i3.

Type de système : Windows 7 Edition Intégrale.

Mémoire Vive : 4 Go.

Disque Dur : 500 Go.

2.2. L'environnement logiciel

Eclipse est un projet, décliné et organisé en un ensemble de sous-projets de développements logiciels, de la fondation Eclipse visant à développer un environnement de production de logiciels libre qui soit extensible, universel et polyvalent, en s'appuyant principalement sur Java.

Son objectif est de produire et fournir des outils pour la réalisation de logiciels, englobant les activités de programmation (notamment environnement de développement intégré et Framework).

2.3. Langage java

Nous avons utilisé comme langage de programmation : le langage orient objet « java ».

Le choix de langage java présente les avantages suivants :

1. C'est un langage bien connu et largement répandu. Il existe de nombreuses librairies qui facilitent le développement des applications.
2. Les compilateurs java sont gratuits.
3. Java permet de définir facilement des interfaces graphiques agréables à utiliser.

3. Modules de l'application

Notre travail consiste en trois principaux modules qui ont été réaliser dans l'application ces trois modules sont :

- Générer les donner.
- Réduire les données.
- Extraction des itemset fréquents.

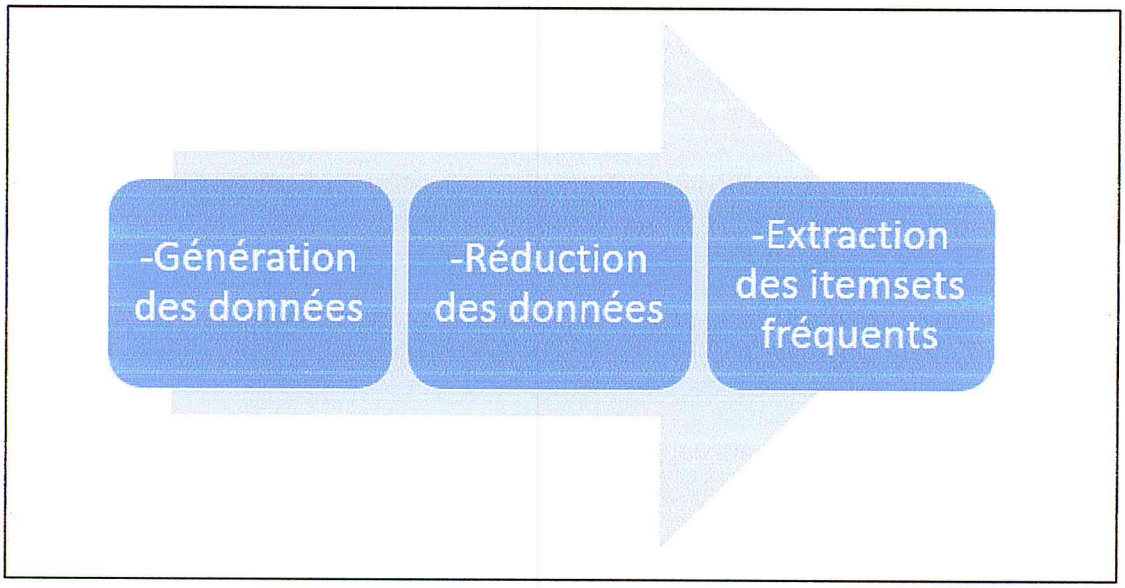


Figure 5.1 : Les trois principaux modules.

3.1. Module de génération des données :

Dans ce module on présentera notre générateur de données évidentielles l'interface suivante nous montre ce générateur et ses paramètres d'entrées ou l'utilisateur doit remplir les quatre champs qui représente le nombre de transactions, le nombre d'objets, le nom des objets et le nom des itemsets pour chaque objets.

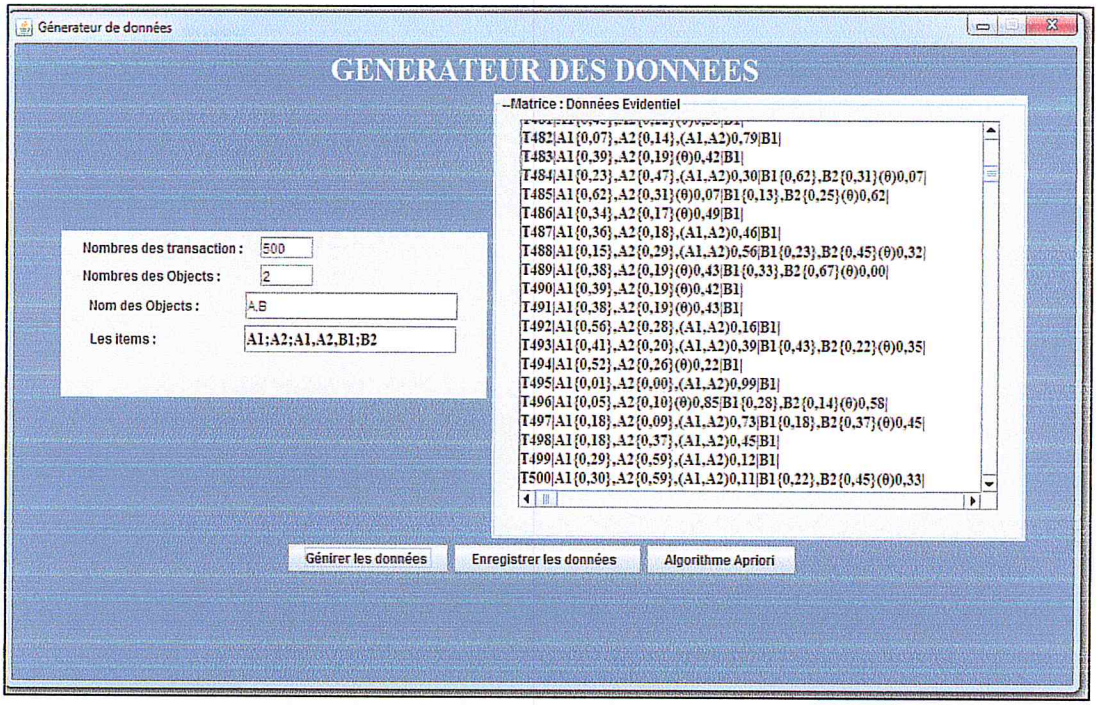


Figure 5.2 : L'interface de module générateur de données.

3.1.1. Scénario

Premièrement l'utilisateur doit remplir les champs dans l'interface. On a choisie comme un jeu d'essai pour les quatre champs, 50 transactions et elle peut être augmenté, 2 objet A et B et 5 itemsets A1 ; A2 ; A1, A2 ; B1 ; B2 les itemsets sont séparer par un point-virgule.

Apres la génération de données l'utilisateur peut choisir d'enregistrer ces données dans un fichier externe pour une utilisation ultérieure en cliquant sur le bouton ***enregistrer les données.***

Un message sera affiché pour informer l'utilisateur s'il veut vraiment commencer la procédure d'enregistrement. S'il appui sur oui les données seront afficher ou chaque itemset est associe par sa masse a l'objet et la transaction auquel il appartient.

Le bouton ***RDS-Apriori*** nous donne la possibilité de poursuivre la procédure et exécuter l'algorithme R-DS-Aprirori. Ce qui va le projeter dans les modules suivant c'est-à-dire la réduction et l'extraction.

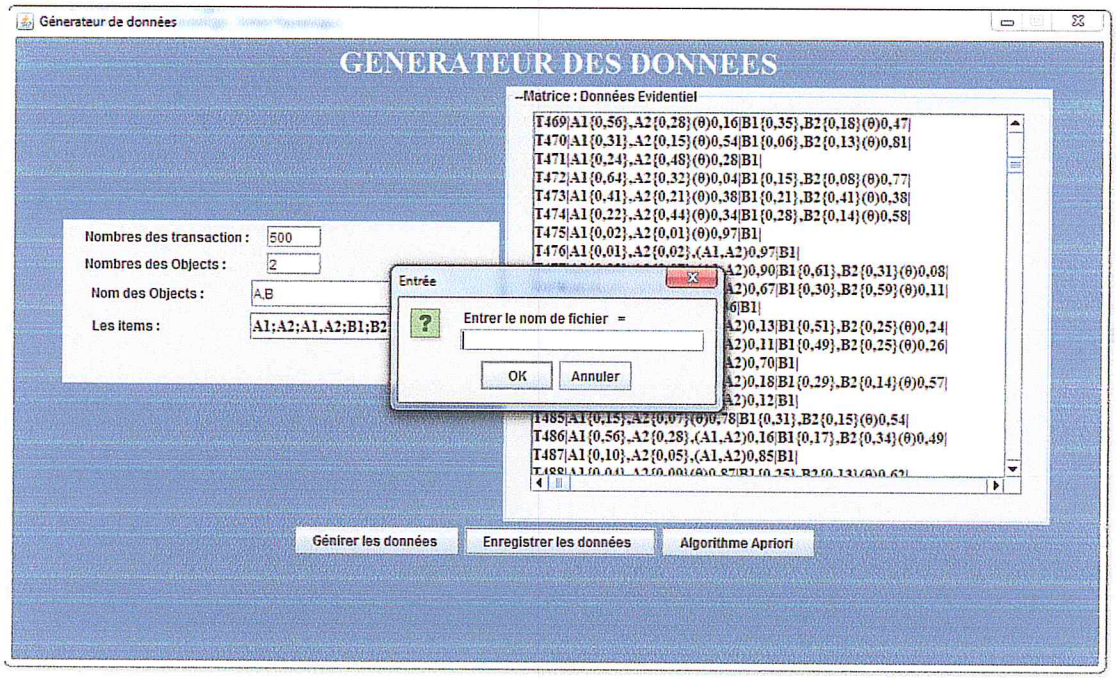


Figure 5.3 : Confirmation de l'enregistrement des données générées.

Remarque : dans l'étape d'enregistrement de donnée on doit écrit le nom de fichier et leur format que ce soit **.csv** ou **.txt**

3.2. Module de réduction des données

Dans cette phase l'utilisateur doit importer les données que cela soit du générateur directement ou bien d'un fichier qui a été enregistré au préalable.

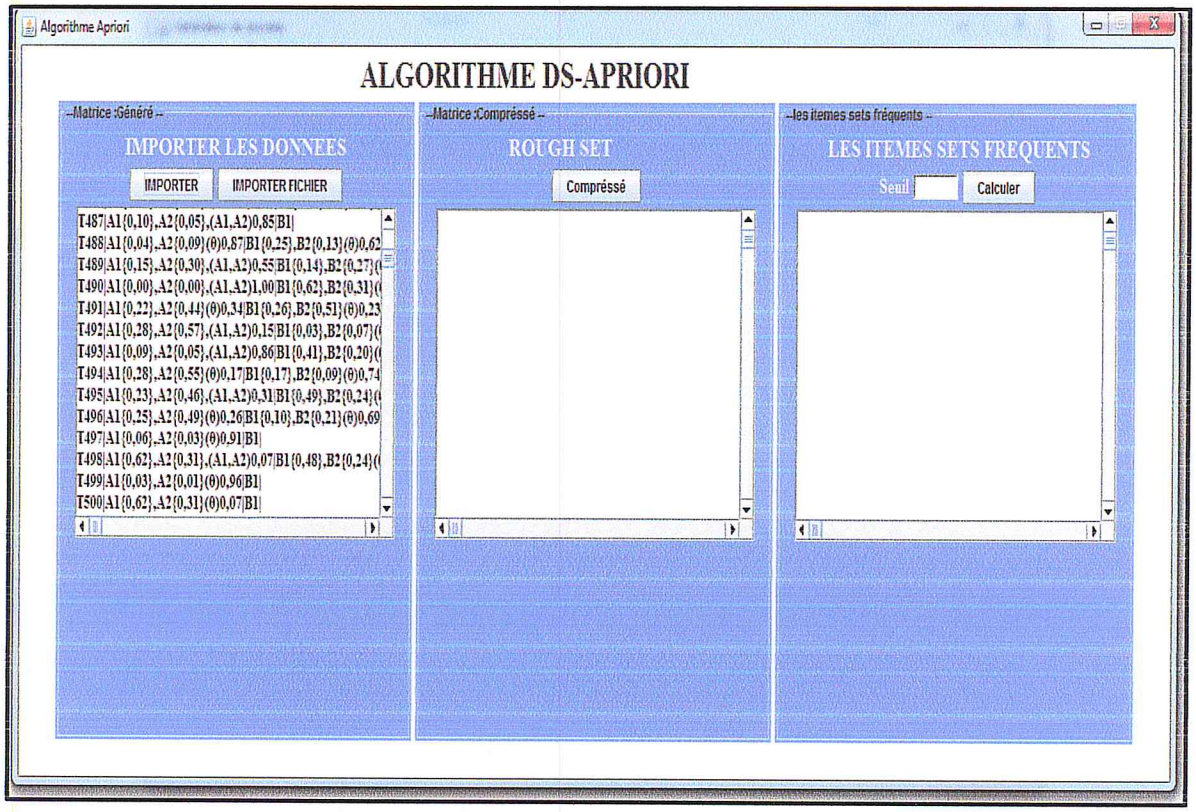


Figure 5.4 : Importation des données.

Apres l'importation des données l'utilisateur peuvent exécuter le deuxième module qui est bien évidemment la réduction ou il va diminuer le nombre des transactions. Comme expliquer dans le précèdent chapitre cette réduction des transactions repose sur la probabilité d'existence et d'ignorance de chaque itemset dans une transaction donnée. Le tableau 5.5 ne montre le résultat de réduction de données.

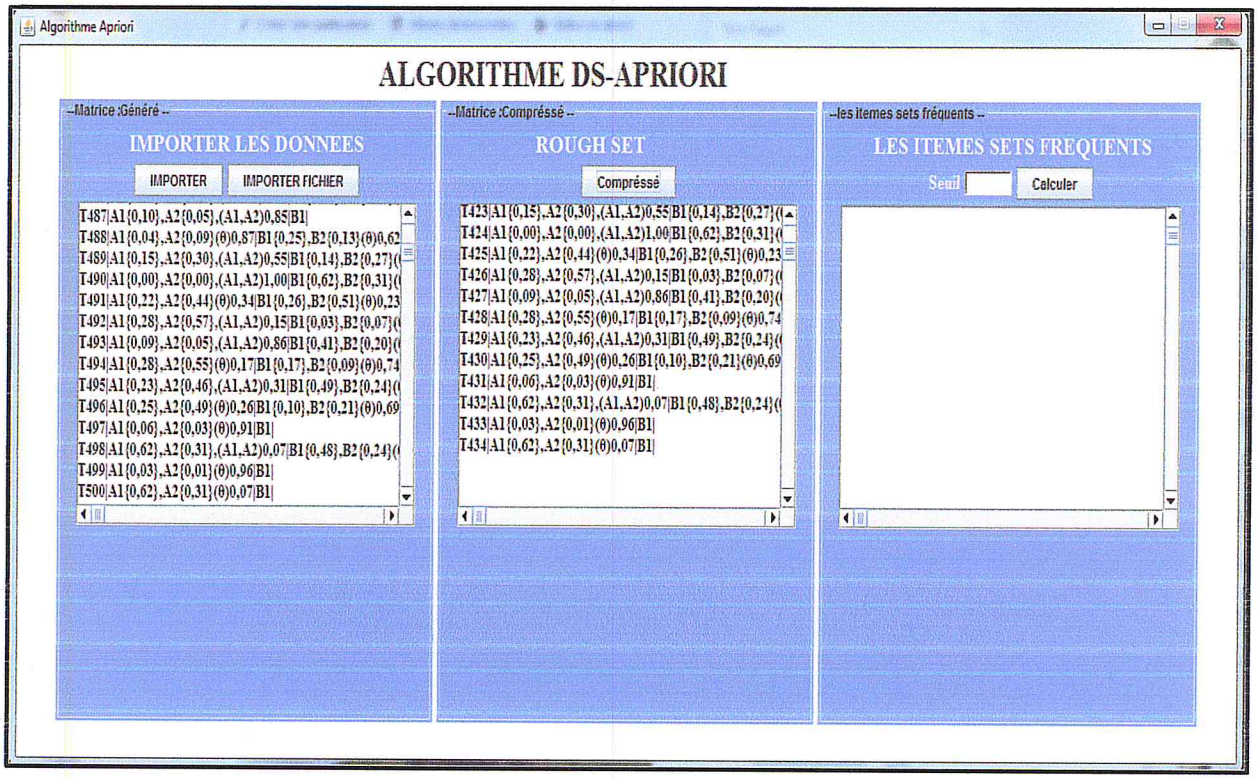


Figure 5.5 : Résultat de module réduction de données.

Dans ce jeu d'essai ou nous avons générer initialement 500 transactions après réduction nous avons eu 434 transaction c'est-à-dire plus de 10% de nos données ont été réduite.

Apres cette phase l'utilisateur peut passer au dernier module et le plus important qui est l'extraction des itemsets fréquents.

3.3. Module d'extraction des itemsets fréquents

Ce module représente le cœur de notre travail ou nous allons appliquer l'algorithme Apriori pour faire l'extraction mais selon une mesure de calcule bien précise qui est l'évidentiel precise support.

Le résultat de ce module est clairement des itemsets fréquent pour chaque itération de l'algorithme Apriori.

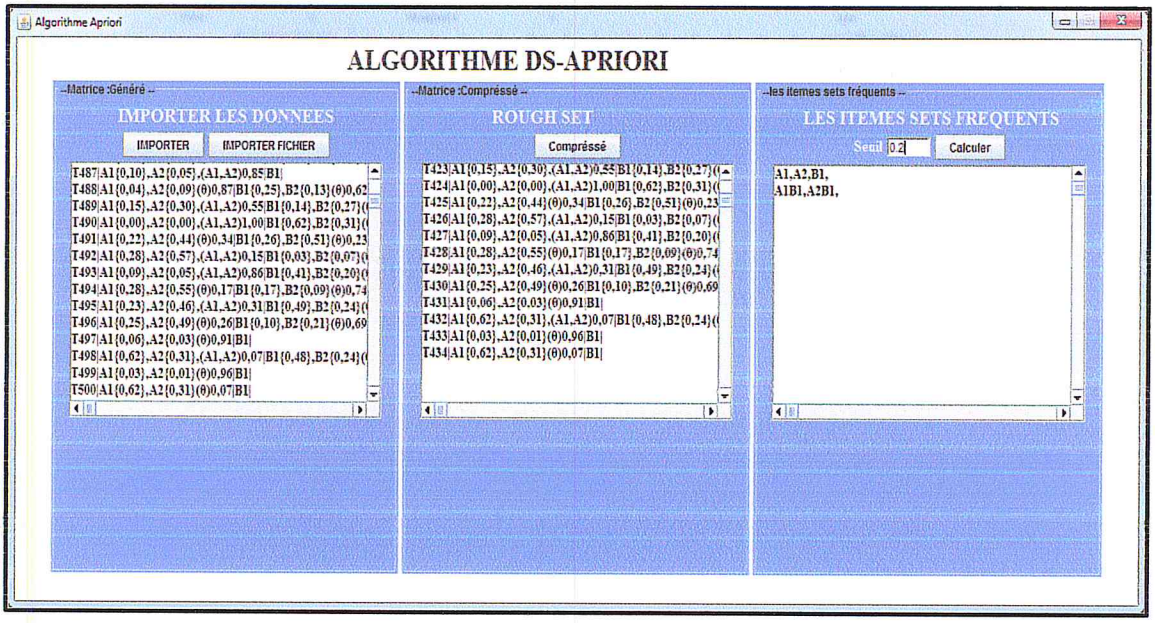


Figure 5.6 : Phase d'extraction des itemsets fréquents.

Comme la figure 5.5 la montre l'utilisateur doit entrer un seuil de tolérance pour les itemsets fréquents puis appuyé sur le bouton calculer et le résultat s'affichera. Par cette action la dernière phase se terminera.

4. Test et comparaison des résultats

4.1. Comparaison par rapport aux données

Maintenant on va faire un jeu d'essai sur la même base de données mais cette fois sans faire de réduction puis on fera une comparaison par rapport au itemset fréquents obtenu comme résultat sur les mêmes données mais une fois sans réduction et une autre fois avec réduction, cela en gardant toujours le même seuil = 20% .

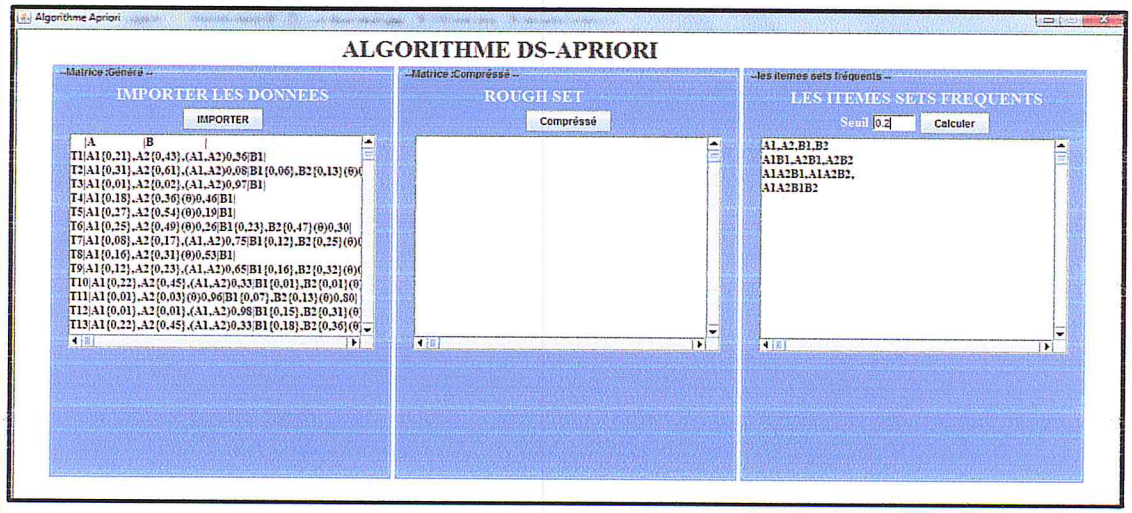


Figure 5.7 :L'affichage des itemsets fréquents.

La figure 5.7 montre l'affichage des itemsets extrait directement à partir du classeur ou bien importer directement à partir du générateur sans passée par le module de réduction.

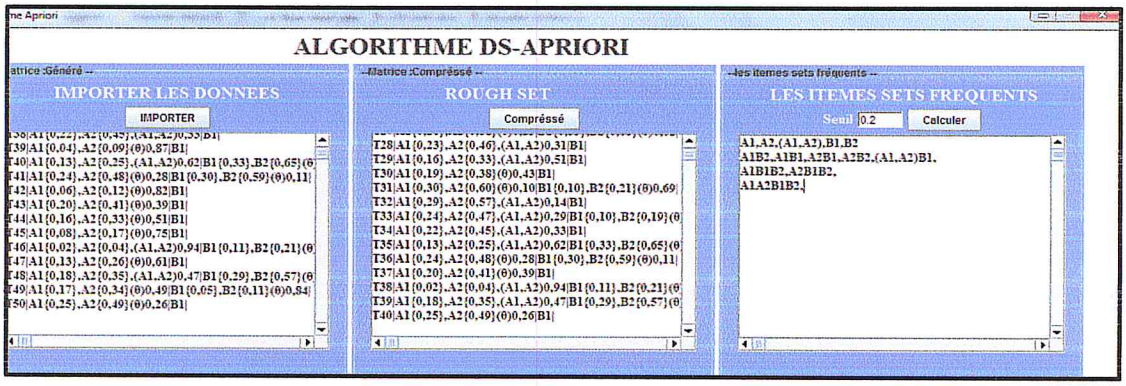


Figure 5.8 : L'affichage des itemset fréquents.

Dans la figure 5.8 l.es itemsets fréquent sont extraire appris l'exécution de deuxième module c a d la réduction des donnée.

Dans la figure 5.7 nous remarquons que les itemsets fréquents sont pauvre contrairement à la figure 5.8 de ces deux résultat on déduit que les données sans prétraitement n'aboutissent pas à un résultat fiable par contre quand on applique la procédure de réduction on réduit en grande quantités les données incomplètes les données deviennent plus fiables et précis.

4.2. Comparaison par rapport au seuil

Maintenant on va faire une comparaison par rapport à l'itemsets fréquent obtenu dans le tableau 5.1 en modifiant le seuil de 20% à 50%.

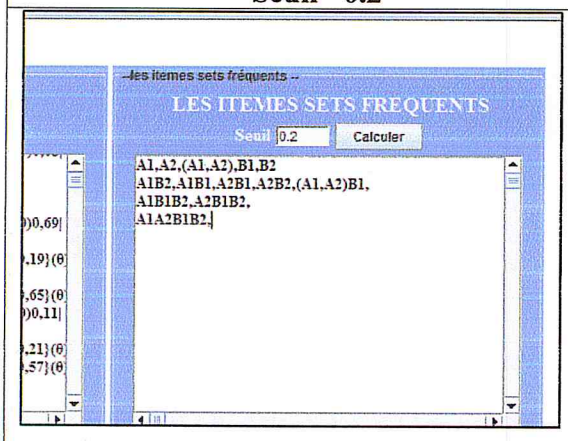
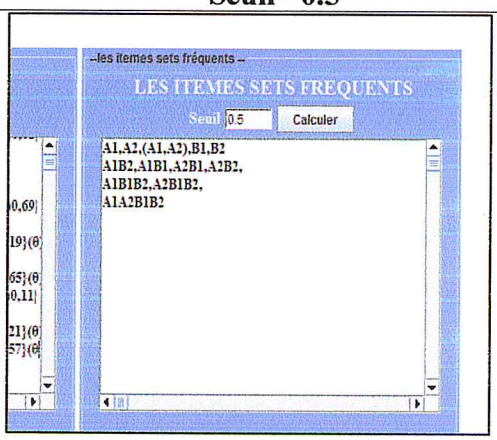
Seuil =0.2	Seuil =0.5
	

Tableau 5.1: Comparaison des résultats par rapport à seuil.

Figure 5.9 : la fréquence des itemsets par rapport au seuil.

Interprétation : on remarque que quand le seuil est petit les itemsets fréquents sont nombreux contrairement à un grand seuil nous avons montré cette fréquence dans une courbe comme montrer dans la figure 5.9.

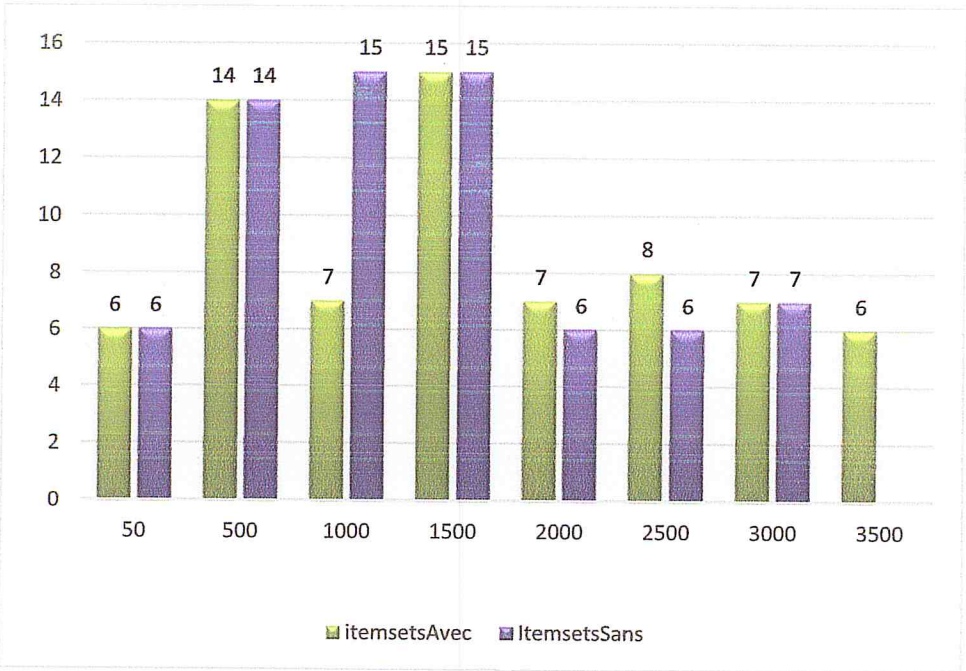


Figure 5.10 : Comparaison du résultat obtenu par rapport à l'extraction avant et après réduction des données.

Interprétation : dans ce graphe on compare l'extraction des itemsets fréquent avant la réduction des données et après réduction des données ou on remarque que le résultat est le même dans la plupart des cas on peut dire qu'il n y a pas une perte d'informations

quand on réduit les données de plus après élimination des données manquants l'extraction devient de qualité ou on remarque de nouveaux itemsets fréquents.

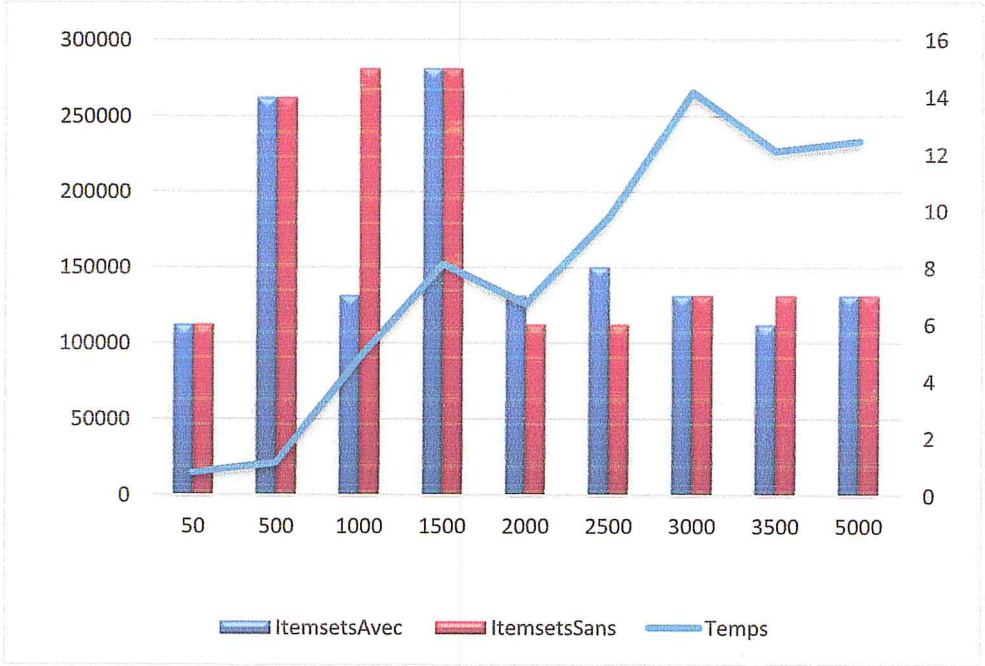


Figure 5.11 : Comparaison du temps d'exécution obtenu par rapport à l'extraction avant et après réduction des données.

Interprétation : la réduction nous a permis d'optimiser le temps d'exécution considérablement en ce qui concerne l'extraction, contrairement à l'extraction sans réduction.

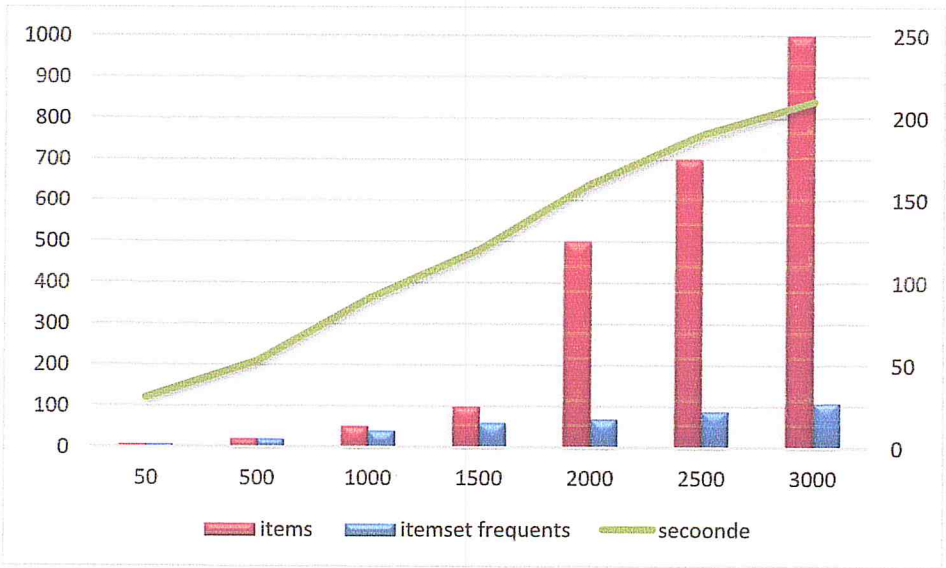


Figure 5.12 : Comparaison du temps d'exécution obtenu par rapport aux items en entrer.

Interprétation : on remarque que le nombre d'itemsets obtenus en résultat dépend du nombre d'items entré en paramètre ou à chaque fois qu'on augmente le nombre des items, les itemsets fréquents obtenu en résultat augmente le contraire est vrai.

5. conclusion

Dans ce chapitre, nous avons décrit le processus de réalisation de notre application en spécifiant l'environnement de développement. Et les étapes pour l'extraction des itemset fréquents nous avons appuyé le résultat obtenu par quelques tests et comparaisons.

« Conclusion Générale»

1. Conclusion

L'extraction des itemsets à partir de données est une technique qui utilise une variété d'outils d'analyse de données pour découvrir, Des motifs fréquents et des relations cachés très intéressantes et qui peuvent être utilisées dans différents domaines pour faire des prédictions valides et / ou prendre des décisions. Cependant cette analyse de données est basée sur leur nature. Les données sont probablement extraites du monde réel et porte dans la plupart des cas des imperfections et donc sont devenues des données imparfaites. C'est pourquoi un grand problème s'est posé qui est l'extraction des itemsets fréquents à partir de données imparfaites.

L'extraction d'itemsets fréquents est une méthode de traitement de données. Pour comprendre et cerner la problématique de l'extraction d'itemsets fréquents à partir de données, nous avons consacré le premier chapitre à présenter la nature de ces données qui est dans notre cas imparfaites, Et montrer les grands types d'imperfections sur les qu'elles nous avons travaillé. Dans le deuxième chapitre nous avons donné un état de l'art sur l'EIF des données parfaites pour pouvoir introduire dans le troisième chapitre les travaux fait dans l'EIF à partir de données imparfaites plus précisément les travaux faits sur les données incertaines et imprécises en ce qui concerne l'incomplétude nous avons ajouté notre propre contribution, Comme prouver dans le quatrième chapitre ou nous avons créé notre propre générateur de données évidentielles puis nous avons fait l'extractions des itemsets fréquents à partir de données imparfaites tout en prenant en considération les types d'imperfections en basant notre travail sur deux grandes théories (théorie d'évidence et la théorie des ensembles d'approximations rough set) qui vont nous aider à faire un traitement sur les imperfections de données respectivement incertaines et incomplètes pour cela un nouvel algorithme a vu le jour basé sur notre travail et nommé Rough-DS-Apriori puis le dernier chapitre comme son nom le montre nous a servi à faire des tests pour valider notre approche proposée.

2. Perspectives

Les perspectives de ce travail préliminaire sont nombreuses. Tout d'abord, nous devons valider la démarche proposée avec des bases de données réelles.

Il est probablement intéressant de pouvoir créer des méthodes qui lancent plusieurs machines sur plusieurs bases de données en même temps pour extraire des itemsets fréquents à partir des données imparfaites dans le contexte de big data.

Tester par la suite les performances du travail accomplit sur les algorithmes de type divisé pour régner.

Utiliser le concept de la théorie rough set comme nous l'avons utilisé en ajoutant le calcul des poids pour avoir un support pondéré.

Bibliographies:

[01] :Vaughn, R. B, J. Farrell, R. Henning, M. Knepper, et K. :Fox Sensor fusion and automatic vulnerability analysis. In Proceedings of the 4Th International Symposium on Information and Communication Technologies, Cape Town, South Africa, pp. 230–235 (2005).

[02] : B.Sarah , J.Martel. :Le choix d'un langage de modélisation des imperfections de l'information en aide à la décision. (2010)

[03] : D.Shafer, G. A: mathematical theory of evidence. Princeton University Press. (1976).

[04] : L.Zadeh. :Fuzzy sets. Information and Control, 3 :338–353, 1965.

[05] : C.Weng , Y.Chen: Mining fuzzy association rules from uncertain data, springerverlag new york, inc. new york, ny, usa issn: 0219-1377 doi. knowledge and information systems 23, 129–152 (2010).

[06] :Z. Pawlak :Rough sets. 11(5) :341–356, 1982.

[07] : Z.Pawlak:Rough sets. International Journal of Computer and Information Sciences, 11:341{356, 1982}.

[08] : Z.Pawlak : Rough Sets. Theoretical Aspects of Reasoning about Data. Kluwer Acad., 1991.

[09]: T.Tanagra :frequent itemsets ,3.10.11, Octobre 2011.

[10]: R. Lovin, : Mining Frequent Patterns, Avril 2012.

[11] : Z.-H. Zhou, H. Li, Q. Yang :PAKDD 2007, LNAI 4426, 2007.

[17] : K. Tannir :Apriori blog Khaled Tannir Dernière consultation , Avril 2017.

[18] : N. Dalvi, D. Suciu :E_cient query evaluation on probabilistic databases The VLDB Journal, 2007.

[19]: M.A,Bach Tobji, B.Ben Yaghlane, K.Mellouli,: Incremental maintenance of frequent itemsets in evidential databases. In: Sossai, C., Chemello, G. (eds.) ECSQARU 2009. LNCS, vol. 5590, pp. 457–468. Springer, Heidelberg (2009).

[20] : Hewawasam, K.K.R., Premaratne, K., Shyu, M.-L., Subasingha, S.P.: Rule mining and classification in the presence of feature level and class label ambiguities. In: SPIE 5803, Intelligent Computing: Theory and Applications III, p. 98 (2005).

[21] : C. Chui, B. Kao, and E. Hung :Mining frequent itemsets from uncertain data: Knowl. Discov. data Min., 2007.

[22] : C. Chui and B. Kao :A decremental approach for mining frequent itemsets from uncertain data Adv. Knowl. Discov. Data Min., 2008.

[23] : C. Leung, M. Mateo, and D. Brajczuk : A tree-based approach for frequent pattern mining from uncertain data Adv. Knowl. Discov, 2008.

[24] : C.Aggarwal, Y. Li, J. Wang, and J. Wang :Frequent pattern mining with uncertain data Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD '09', p. 29, 2009

[25] : C. Leung and S. Tanbeer :Fast tree-based mining of frequent itemsets from uncertain data , Database Syst. Adv. Appl., 2012.

[26] : T. Calders, C. Garboni, and B. Goethals: Efficient pattern mining of uncertain data with sampling, Adv. Knowl. Discov., 2010.

[27] : B. Budhia, A. Cuzzocrea, and C. Leung,: Vertical Frequent Pattern Mining from Uncertain Data., KES, 2012.

[28] : T. Bernecker, H. Kriegel, M. Renz, and F. Verhein, :Probabilistic frequent pattern growth for itemset mining in uncertain databases, Sci. Stat., 2012.

[29] : C. Leung and S. Tanbeer, : PUF-tree a compact tree structure for frequent pattern mining of uncertain data, Adv. Knowl. Discov. Data, 2013.

[30] : A. Samet and T. Dao :Mining over a Reliable Evidential Database Application on amphiphilic chemical database,pp. 1257–1262, 2015.

[31] : A. Samet, E. Lefèvre, and S. Yahia: Mining frequent itemsets in evidential database,Knowl. Syst. Eng., 2014.

[32]: M. Tobji, B. Yaghlane, and K. Mellouli :Frequent itemset mining from databases including one evidential attribute, Scalable Uncertain. Manag., 2008.

[33] M. Tobji, B. Yaghlane, and K. Mellouli: A new algorithm for mining frequent itemsets from evidential databases,:Proc. IPMU, 2008.

[34] I. Bloch : Fuzzy Sets and Possibility Theory, Inf. Fusion Signal Image Process.(2010)

[35] : A.Samet, , E.Lefevre, S.Ben Yahia : Mining frequent itemsets in evidential database. In: Proceedings of the Fifth International Conference on Knowledge and Systems Engeneering, Hanoi, Vietnam, pp. 377–388 (2013)

[36] : M.A, BachTobji, B, Benghlane, Extraction des itemsets fréquents à partir de données évidentielles : Application a une base de données éducationnels 2015.

[37]: Nguyen Do Van: Rough Set Models and Knowledge Acquisition for Imperfect information systems , p[64,79] 2014