Chapitre 3 : Les règles d'association

1. **Introduction :** certaines données transactionnelles renferment des connaissances sous forme d'associations entre divers objets de différents types mais appartenant aux mêmes transactions. Par exemple, le panier des achats peut contenir des articles de types différents (lait, pain, beurre). L'analyse et la découverte des associations de types « si un client achète du lait alors il achète du pain » ou la découverte des articles associés peut être utile à différentes fins, telles que la réorganisation des rayons, les promotions, etc. Cependant, deux problèmes se présentent lors de la recherche de ces associations.

    a. Le volume de données peut rendre la découverte d'association coûteuse.
    b. Certaines associations peuvent se produire par hasard et non pas d'utilité.

2. **Exemple Illustratif** : On prend l'exemple suivant des achats concernant les articles: pain, lait, jus, beurre, œufs, cola (voir tableau). On fait abstraction des quantités. Une transaction est matérialisée par un ticket de caisse. Chaque achat peut être vu comme un objet et chaque article comme une variable. Dans ce cas, les variables sont binaires car elles peuvent avoir la valeur 1 si l'article est acheté et 0 sinon. On peut mettre les achats sous la forme suivante.

ID ticket	Pain	Lait	Jus	Beurre	Œufs	Cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	0

Dans le tableau ci-haut, on désigne par

- **Ensemble d'articles (*itemset*) :** ensemble d'articles qui sont achetés ensemble. Ex : dans la transaction 1, le itemset contient du pain et du lait. Deux transactions ou plus peuvent être identiques.
- **Règle d'association :** l'implication *Si A alors B* où A et B désigne deux sous-ensembles disjoints d'articles de l'ensemble total.

3. **Mesures d'évaluation des règles d'association :** la découverte des associations peut résulter en plusieurs combinaisons d'articles qui ne sont pas toutes utiles car s'agissant d'associations rares ou hasardeuses. Pour garder les bonnes associations, on utilise deux critères d'évaluation. Soit A et B deux ensembles d'articles :
    - *Support d'une règle d'association* : il s'agit du rapport entre le nombre de transactions qui contiennent A et B sur le nombre total de transactions.
    - *Confiance accordée à une règle*: il s'agit du rapport entre le nombre de transactions qui contient A et B sur le nombre de transactions qui contiennent A.

⇒ Une bonne règle est celle ayant un support et une confiance élevés.

Exemple : soit la règle si achat du Pain alors achat de Lait.

- Support : 3 (nombre de transactions contenant le pain et le lait) /5= 60%
- Confiance : 3 (nombre de transactions contenant le pain et le lait) / 4 (nombre de transaction contenant du pain) = 75%.

4. Démarche de d'extraction des règles d'association

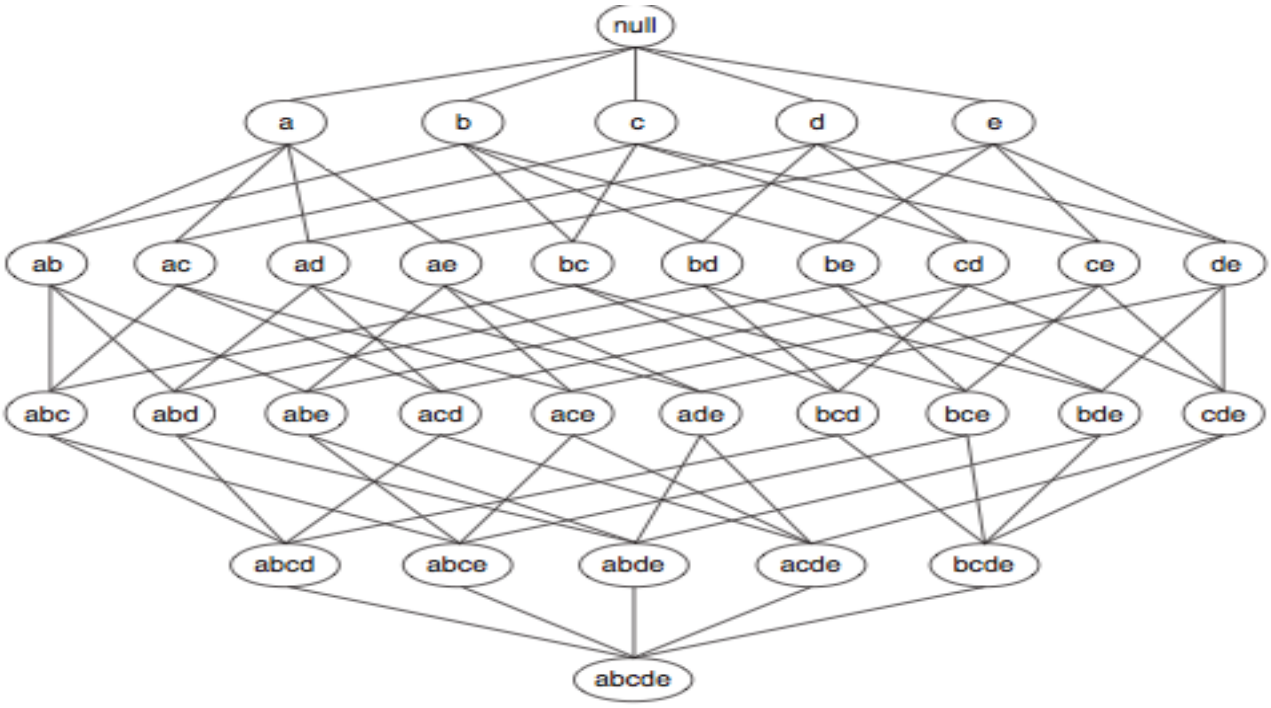
a. ***Bien choisir les item et les niveaux :*** les items à choisir correspondent à des produits dans une supérette ou à autre chose dans d'autres domaines. Dans le cas des produits par exemple, il faut sélectionner les produits (pain, beurre, viande) ou leurs catégories (produits laitiers, vêtements, etc.), comme on peut mélanger différents niveaux hiérarchiques des produits selon le besoin d'analyse.

b. ***Fixer un degré d'exigence sur les règles :*** lorsque le volume de données est suffisamment grand, le nombre de règles peut également être grand et leur calcul fastidieux il faut fixer les deux paramètres **support** et **confiance** pour aboutir à des règles de qualité d'une part et limiter le nombre de règles produites d'autre part. Ce support et cette confiance sont **minimaux**.

c. ***Rechercher les itemsets fréquents :*** ils correspondent aux itemsets ayant un support supérieur ou égal au support minimal.

d. ***Produire les règles d'associations :*** à partir des itemsets fréquents, on garde les règles ayant une confiance supérieure ou égale à la confiance minimale.

5. **Génération des itemsets fréquents :** on peut se baser sur un treillis d'itemsets pour recenser tous les cas possibles. La figure suivante illustre le treillis de 5 produits.



La génération des itemsets fréquents se fait par le test de présence de chaque itemset dans les transactions, ce qui peut devenir complexe devant un grand nombre de transactions. Pour réduire la complexité, on procède à l'élagage du treillis en se basant sur le support.

- ***Elagage par support (support-based pruning)*** : il consiste à réduire le treillis en se basant sur le principe suivant : *Si un itemset est fréquent (support supérieur ou égal au support minimum) alors tous ses sous-itemsets sont aussi fréquents*. Inversement, *si un itemset n'est pas fréquent, tous ses super-itemsets ne sont pas fréquents*.

- ***Algorithme Apriori*** : il s'agit de l'un des premiers algorithmes de recherche des règles d'associations. L'algorithme commence par l'énumération des itemsets de cardinalité 1. Les itemsets fréquents sont ceux ayant un support égal ou supérieur au support minimum. A partir de ces itemsets fréquents, on génère les itemsets de cardinalité 2 et on ne garde que les fréquents. On procède ainsi jusqu'à ce que l'on ne puisse plus générer d'itemsets. La figure suivante (Tan et al., 2006) représente le pseudo code de l'algorithme Apriori.

1: $k = 1$.
2: $F_k = \{ i \mid i \in I \wedge \sigma(\{i\}) \geq N \times minsup\}$. {Find all frequent 1-itemsets}
3: **repeat**
4: $k = k + 1$.
5: $C_k = \text{apriori-gen}(F_{k-1})$. {Generate candidate itemsets}
6: **for** each transaction $t \in T$ **do**
7: $C_t = \text{subset}(C_k, t)$. {Identify all candidates that belong to t}
8: **for** each candidate itemset $c \in C_t$ **do**
9: $\sigma(c) = \sigma(c) + 1$. {Increment support count}
10: **end for**
11: **end for**
12: $F_k = \{ c \mid c \in C_k \wedge \sigma(c) \geq N \times minsup\}$. {Extract the frequent k-itemsets}
13: **until** $F_k = \emptyset$
14: Result $= \bigcup F_k$.

où F_K désigne un itemset de cardinalité K et C_K désigne les itemsets candidats avant le prunning, *T* désigne l'ensemble de transactions et *t* désigne une transaction.

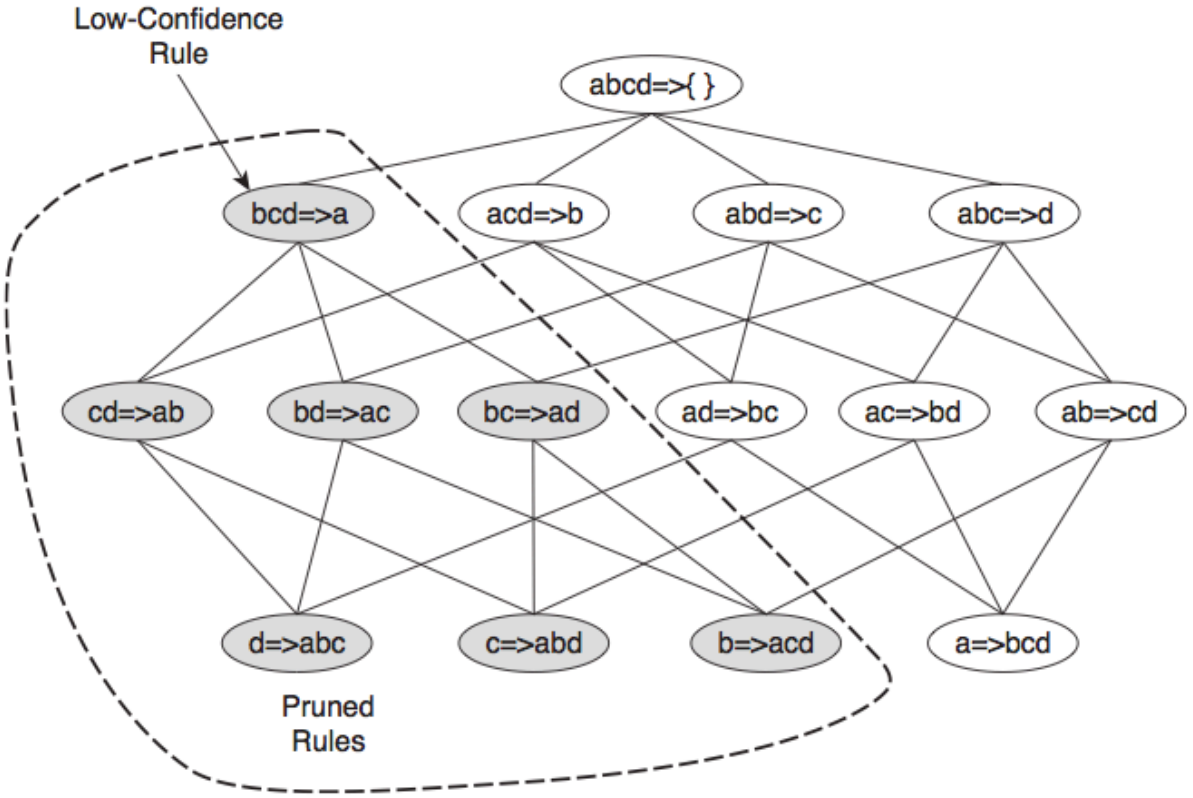
- <u>Génération des itemsets candidats</u> : il existe différentes manières de générer les itemsets fréquents (étape 5 de l'algorithme). La procédure apriori-gen(F_{K-1}) pour générer les itemsets fréquents candidats (C_K) de cardinalité K à partir des itemsets fréquents de cardinalité K-1 est basée sur la fusion des ensembles F_{K-1} ayant les K-2 premier articles identiques et le (K-1)^{ème} article différent. Par exemple, on fusionne les ensembles (a, b, c) et (a, b, d) en (a, b, c, d) mais pas (a, b, c) et (b, d, e).

6. **Génération des règles :** la génération des règles d'association se fait à partir des ensembles d'items fréquents. Le nombre de règles candidates qu'on peut générer est de 2^K-2 en ignorant les règles avec antécédent ou conséquence nulles. Comme la génération et l'exploration de toutes les règles peuvent également devenir coûteuses, on procède à l'élagage des règles au niveau du treillis. Pour effectuer cet élagage, on se base sur le

théorème suivant.

Théorème : soit Y un itemset fréquent et X un sous-ensemble de Y. Soit X' un sous-ensemble de X. alors *Si la règles X → Y-X ne satisfait pas le seuil de confiance, alors toute règle X'→ Y-X ne peut pas satisfaire le seuil de confiance également.*

- ***Génération des règles dans l'algorithme Apriori :*** la figure suivante (Tan et al. 2006) le treillis généré à partir d'un itemset de cardinalité 4. Les règles en gris représentent les règles élaguées à cause d'une confiance baisse de la règle bcd → a.



La génération des règles se fait de manière incrémentale selon le pseudo algorithme suivant :

1) On prend en entrée chaque itemset fréquent (par exemple, l'ensemble contenant abcd)
   a. A partir de la règle abcd → **Φ**, on génère toutes les règles possibles.
2) Pour chaque règle générée, on teste sa confiance
3) Si elle est inférieure au seuil minimal, la règle est éliminée et aucune génération ne se fait à partir d'elle.
4) Sinon, la règle est maintenue.
5) Reboucler sur l'étape 2.