

Extraction de Motifs : Les règles d'association

HMIN232M

Pascal Poncelet
LIRMM

Pascal.Poncelet@lirmm.fr
<http://www.lirmm.fr/~poncelet>

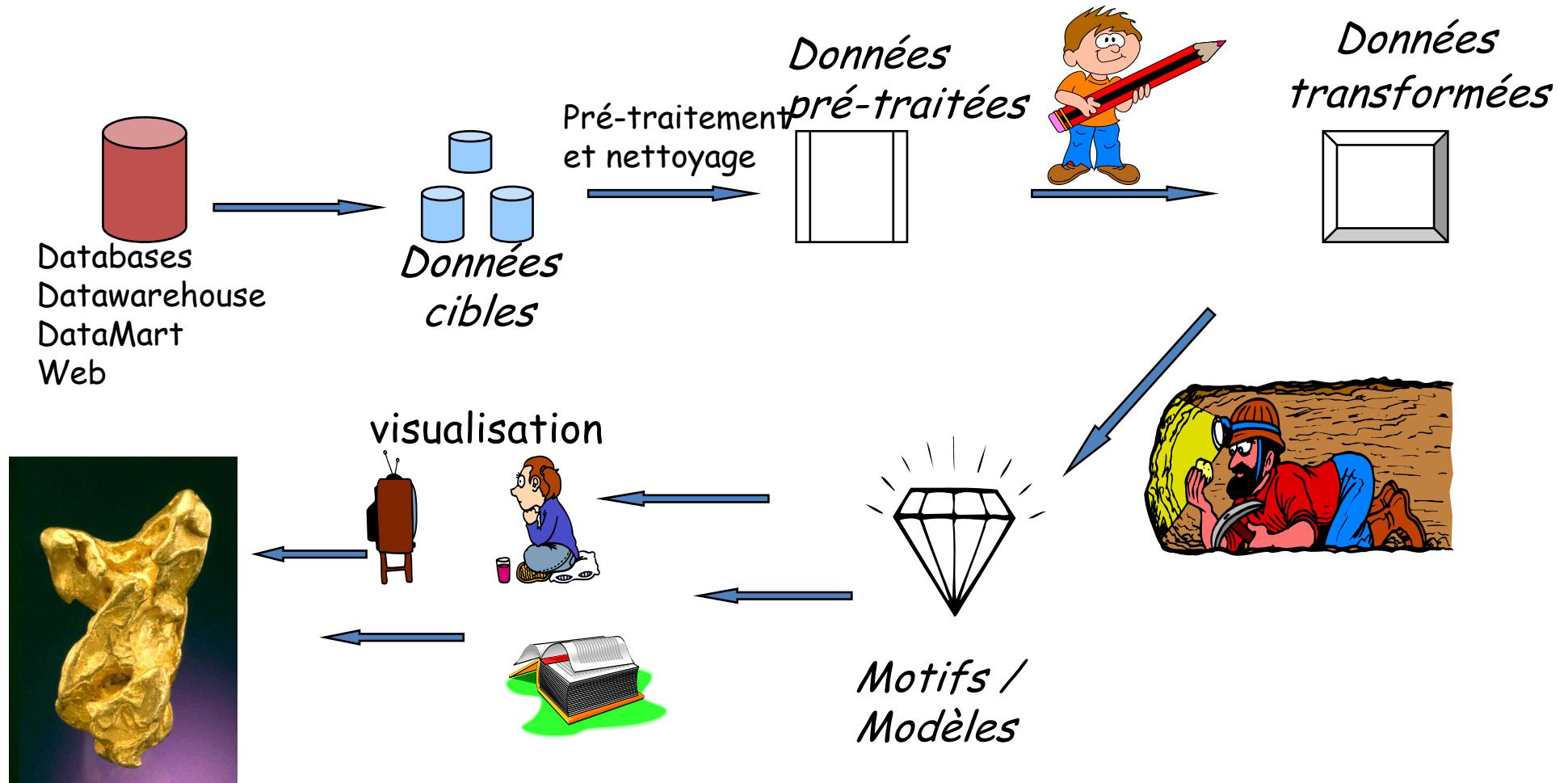


Plan

- Contexte général
 - Règles d'association
 - Un exemple d'application : Web Usage Mining
-
- (la partie suivante est en M2)
 - Motifs séquentiels, trajectoires, motifs complexes
 - Conclusions



Le processus de KDD



Recherche de motifs fréquents

- Qu'est ce qu'un motif fréquent ?
 - Un motif (ensemble d'items, séquences, arbres, ...) qui interviennent fréquemment ensemble dans une base de données [AIS93]
- Les motifs fréquents : une forme importante de régularité
 - Quels produits sont souvent achetés ensemble ?
 - Quelles sont les conséquences d'un ouragan ?
 - Quel est le prochain achat après un PC?



Recherche de motifs fréquents

- **Analyse des associations**
 - Panier de la ménagère, cross marketing, analyse de textes, analyse de gènes,
 - Corrélation ou analyse de causalité
- **Clustering et Classification**
 - Classification basée sur les associations
- **Analyse de séquences**
 - Web Mining, détection de tendances, analyses ADN
 - Périodicité partielle, associations temporelles/cycliques

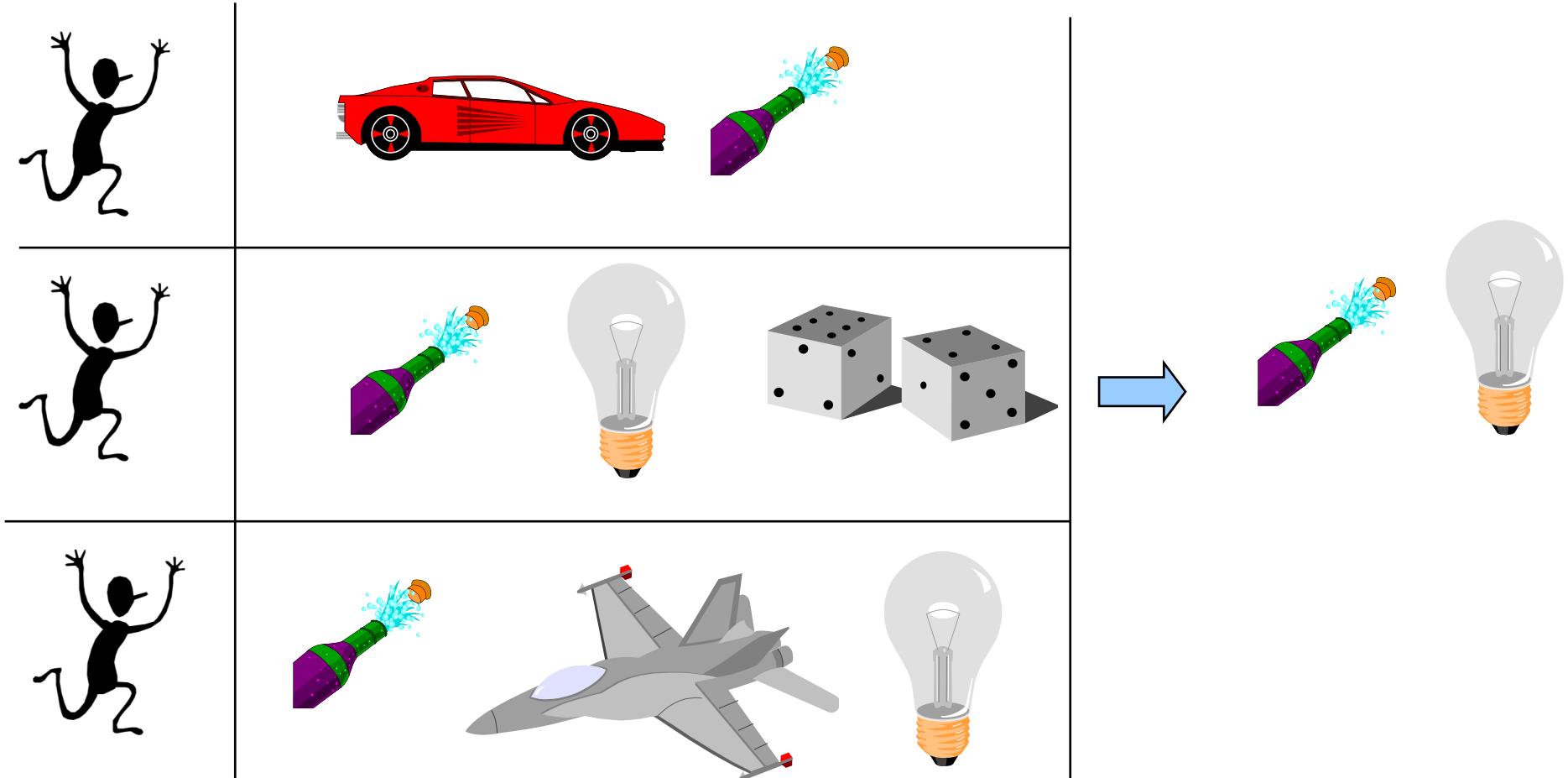


« Panier de la ménagère »

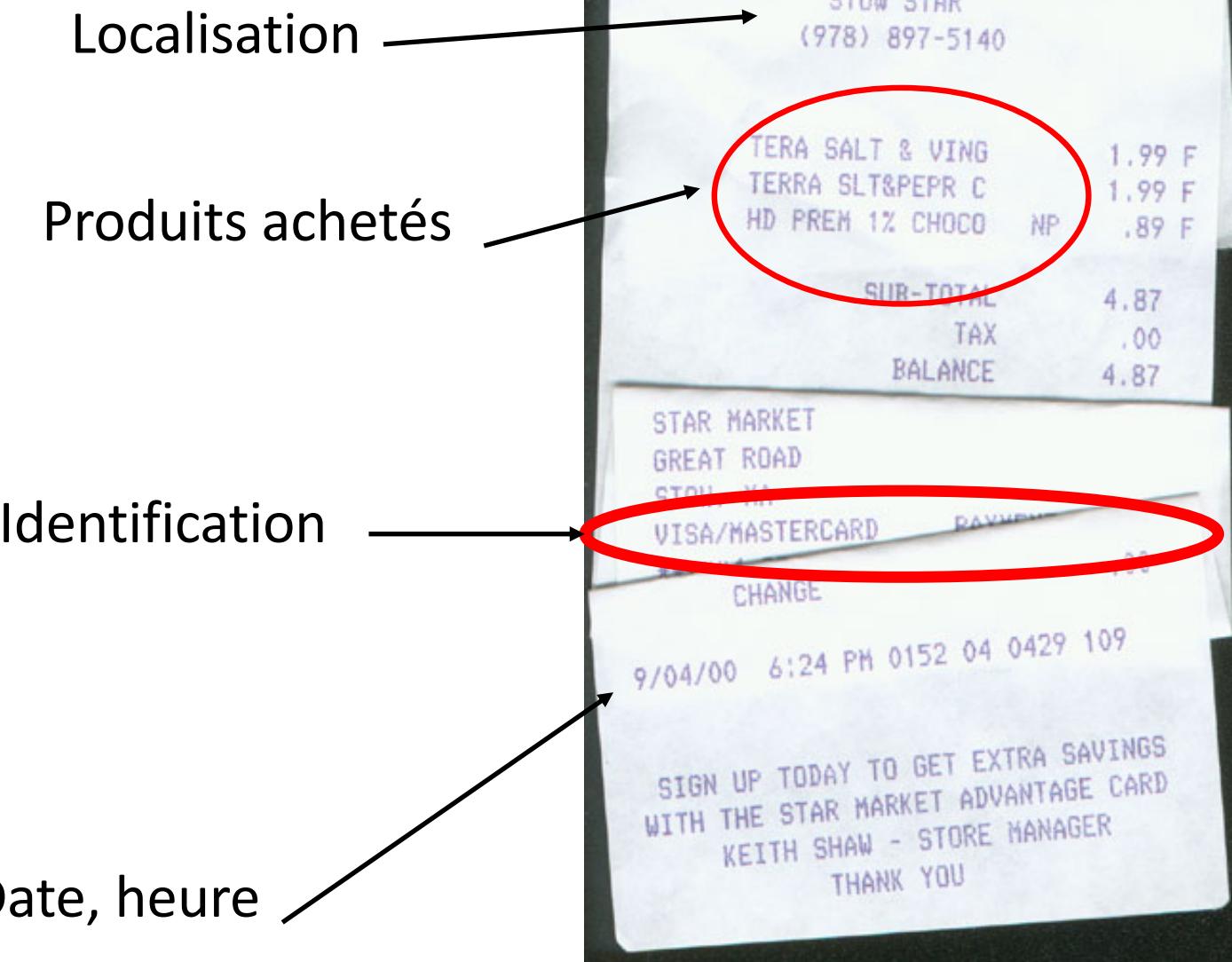
- **Recherche d'associations**
 - recherche de corrélations entre attributs (items)
 - caractéristiques : « panier de la ménagère »
 - de très grandes données
 - limitations : données binaires
- **Recherche de motifs séquentiels**
 - recherche de corrélations entre attributs (items) mais en prenant en compte le temps entre items => comportement



Recherche de règles d'association



Panier de la ménagère



Aidons Mme Guénolé

Epicerie de Mme. Guénolé.

Camembert
Bière
Perrier
Oeufs,
Salade
Confiture
Couches

Biscuit
Crème
Couches
Pain,
Confiture

Couches
Confiture
ièvre
amembert
ait
alade
ain
oudre de cacao
iscuit

Epicerie de Mme. Guénolé.

mbert
er
e
ture,
ies





Epicerie de
Mme. Guénolé.

Biscuit
Crème
Couches
Pain
Confiture



Epicerie de
Mme. Guénolé.

Camembert
Bière
Perrier
Oeufs
Salade
Confiture
Couches



Epicerie de
Mme. Guénolé.

Bière
Lait
Crème
Pain
Salade
Couches
Poudre de cacao
Oeufs



Epicerie de
Mme. Guénolé.

Oeufs
Salade
Pain



Epicerie de
Mme. Guénolé.

Salade
Confiture
Pain
Oeufs
Crème
Perrier
Biscuit



Epicerie de
Mme. Guénolé.

Couches
Confiture
Bière
Camembert
Lait
Salade
Pain
Poudre de cacao
Biscuit



Epicerie de
Mme. Guénolé.

Biscuit
Crème
Oeufs
Pain
Couches
Lait
Poudre de cacao
Confiture



Epicerie de
Mme. Guénolé.

Lait
Bière
Perrier
Oeufs
Couches



Epicerie de
Mme. Guénolé.

Pain
Biscuit
Couches
Lait
Camembert
Bière
Poudre de cacao

La légende

Stories – Beer and Diapers



- ◆ **Diapers and Beer.** Most famous example of market basket analysis for the last few years.
If you buy diapers, you tend to buy beer.
- T. Blischok headed Terradata's Industry Consulting group.
- K. Heath ran self joins in SQL (1990), trying to find two itemsets that have baby items, which are particularly profitable.
- Found this pattern in their data of 50 stores/90 day period.
- Unlikely to be significant, but it's a nice example that explains associations well.

Ronny Kohavi ICML 1998

Recherche de règles d'association

- Règles de la forme

ANTECEDENT → CONSEQUENT [Support, Confiance]

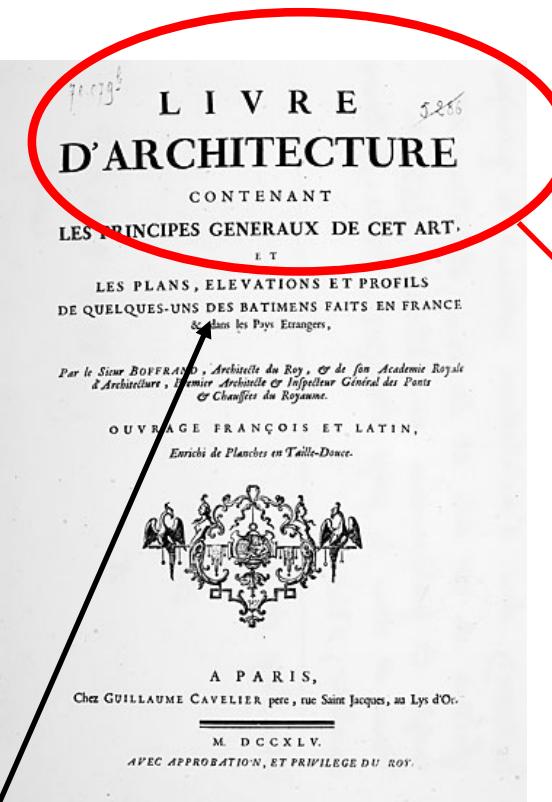
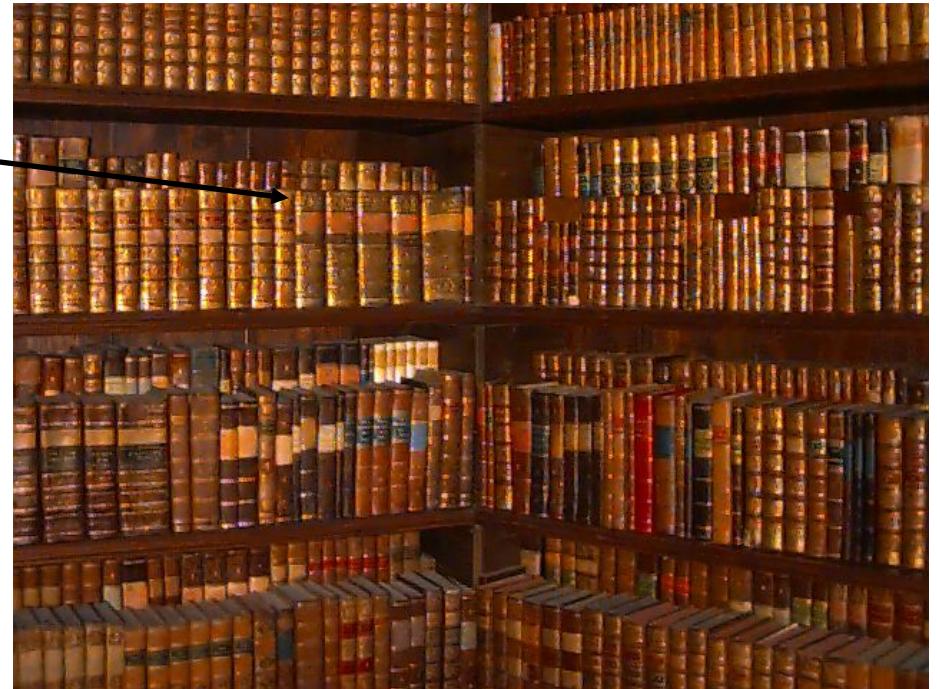
(support et confiance sont des mesures d'intérêt définies par l'utilisateur)

- Achat (x, « Beurre ») ET Achat (x, « Pain ») → Achat(x, « Lait »)
[70%, 80%]
- Achat (x, « Bière ») ET Achat (x, « Gâteaux ») → Achat (x, « Couches ») [30%, 80%]
- Achat (x, « Caviar ») → Achat(x, « Champagne ») [10%, 90%]



Panier de la ménagère

Localisation



Identification

Position # Date

Premier paragraphe

« Livre d'architecture contenant les principes généraux ... »

Mots # Produits

Interprétation

- $R : X \rightarrow Y (A\%, B\%)$
 - **Support** : portée de la règle
Proportion de paniers contenant tous les attributs A% des clients ont acheté les 2 articles X et Y
 - **Confiance** :
Proportion de paniers contenant le conséquent parmi ceux qui contiennent l'antécédent
B% des clients qui ont acheté X ont aussi acheté Y
- Beurre, Pain → Lait [70%, 80%]
- Bière, Gâteaux → Couches [30%, 80%]
- Caviar → Champagne [10%, 90%]



Utilisation des règles d'association

Bière, ... →Couches

- **Couches** comme conséquent
déterminer ce qu'il faut faire pour augmenter les ventes
- **Bière** comme antécédent
quel produit serait affecté si on n'arrête de vendre de la bière
- **Bière** comme antécédent et **Couche** comme conséquent
quels produits devraient être vendus avec la Bière pour promouvoir la vente de couches



Définitions des ensembles fréquents

- Soit un ensemble $I = \{I_1, I_2, \dots, I_m\}$ d'items, une transaction T est définie comme les sous-ensembles d'items dans I ($\subseteq I$).
 - $I = \{\text{Bière, Café, Couche, Gâteaux, Moutarde, Saucisse...}\}$
 - $T_1 = \{\text{Café, Moutarde, Saucisse}\}$
- Une transaction n'a pas de duplicats
- Soit une base de données D un ensemble de n transactions et chaque transaction est nommée par un identifiant (TID).
 - $D = \{\{T_1, \{\text{Café, Moutarde, Saucisse}\}}, \{T_2, \{\text{Bière, Café, Gâteaux}\}\}, \dots\}$



Une base de données

- Une représentation de la base de données D

Client	Pizza	Lait	Sucre	Pommes	Café
1	1	0	0	0	0
2	0	1	1	0	0
3	1	0	0	1	1
4	0	1	0	0	1
5	1	0	1	1	1

- En fait

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$



Définition des ensembles fréquents (cont.)

- Une transaction T **supporte** un ensemble $X \subseteq I$ si elle contient tous les items de X ($X \subseteq T$).
 - T_1 supporte {Café, Moutarde, Saucisse}
- Support de X (**Supp(X)**) : fraction de toutes les transactions dans D qui supportent X .
- Si $\text{supp}(X) \geq s_{\min}$ l'ensemble X est dit **fréquent**.

- Un ensemble d'items (*itemset*) X de cardinalité $k = |X|$ est appelé un *k-itemset*.
 - 3-itemset : {Café, Moutarde, Saucisse}



Propriétés des ensembles fréquents

- **Propriété 1 : support pour les sous-ensembles**
 - Si $A \subseteq B$ pour les itemsets A, B alors $\text{supp}(A) \geq \text{supp}(B)$ car toutes les transactions dans D qui supportent B supportent aussi nécessairement A.
 $A=\{\text{Café, Moutarde}\}$, $B =\{\text{Café, Moutarde, Saucisse}\}$
- **Propriété 2 : les sous-ensembles d'ensembles fréquents sont fréquents**
- **Propriété 3 : les sur-ensembles d'ensembles non fréquents sont non fréquents (anti-monotonie)**



Définition des Règles d'association

- Une règle d'association est une implication de la forme

$$R : X \rightarrow Y$$

où X et Y sont des itemsets disjoints :
 $X, Y \subseteq I$ et $X \cap Y = \emptyset$.

Bière, Gâteaux \rightarrow Couches



Définition des Règles d'association (cont.)

- Confiance (*confidence*) dans une règle R
- Si une transaction supporte X, elle supporte aussi Y avec une certaine probabilité appelée **confiance** de la règle ($\text{conf}(R)$).

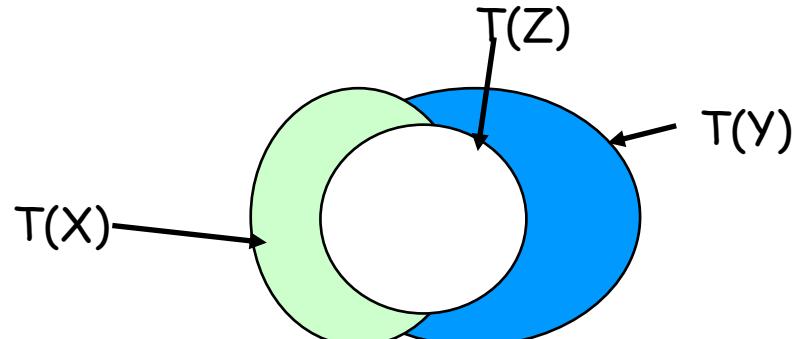
$$\begin{aligned}\text{conf}(R) &= p(Y \subseteq T \mid X \subseteq T) \\ &= p(Y \subseteq T \wedge X \subseteq T) / p(X \subseteq T) \\ &= \text{support}(X \cup Y) / \text{support}(X)\end{aligned}$$

$$\text{conf}(R) = \frac{\text{Supp}(\text{Bière}, \text{Gâteaux}, \text{Couches})}{\text{Supp}(\text{Bière}, \text{Gâteaux})} \geq \text{confiance ?}$$



Propriétés des règles d'association

- **Propriété 4 : pas de composition des règles**
 - Si $X \rightarrow Z$ et $Y \rightarrow Z$ sont vrais dans D , $X \cup Y \rightarrow Z$ n'est pas nécessairement vrai.
 - Considérons le cas où $X \cap Y = \emptyset$ et les transactions dans D supportent Z si et seulement si elles supportent X ou Y , alors l'ensemble $X \cup Y$ a un support de 0 et donc $X \cup Y \rightarrow Z$ a une confiance de 0%.
- **Propriété 5 : décomposition des règles**
 - Si $X \cup Y \rightarrow Z$ convient, $X \rightarrow Z$ et $Y \rightarrow Z$ peut ne pas être vrai.



Propriétés des règles d'association

- **Propriété 6 : pas de transitivité**
 - Si $X \rightarrow Y$ et $Y \rightarrow Z$, nous ne pouvons pas en déduire que $X \rightarrow Z$.
- **Propriété 7 : déduire si une règle convient**
 - Si $A \rightarrow (L-A)$ ne vérifie pas la confiance alors nous n'avons pas $B \rightarrow (L-B)$ pour les itemsets L , A , B et $B \subseteq A$.



En résumé

- Itemsets : A, B ou B, E, F
- Support pour un itemset
 - Supp (A,D)=1
 - Supp (A,C) = 2
- Itemsets fréquents (minSupp=50%)
 - {A,C} est un itemset fréquent
- Pour minSupp = 50% et minConf = 50%, nous avons les règles suivantes :
 - A → C [50%, 50%]
 - C → A [50%, 100%]

Trans. ID	Items
1	A, D
2	A, C
3	A, B, C
4	A, B, E, F



Schéma algorithmique de base

- La plupart des approches utilise le même schéma algorithmique
- Pour construire les règles d'association, le support de tous les itemsets fréquents dans la base doit être calculé
- L'algorithme procède en deux phases :
 - 1) Génération de tous les ensembles fréquents
 - 2) Génération des règles d'association



Comptage des itemsets

- Une première approche
- $I = \{A, B, C\}$
- Génération de tous les cas possibles :
 $\{\emptyset\}, \{A\}, \{B\}, \{C\},$
 $\{A, B\}, \{A, C\}, \{B, C\}$
 $\{A, B, C\}$
- Comptage du support

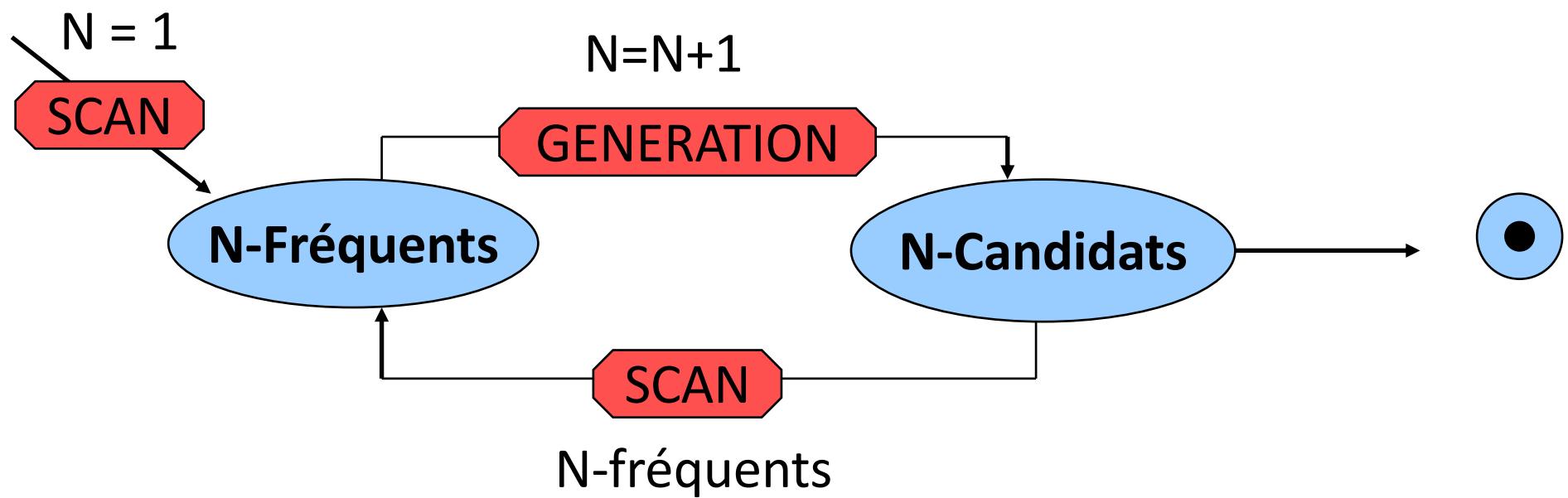


Génération des ensembles fréquents

- Le nombre d 'ensemble fréquent potentiel est égal à la taille du produit cartésien de tous les items qui croit exponentiellement en fonction du nombre d 'items considérés.
- Approche naïve : recherche exhaustive et test de tous les ensemble du produit cartésien pour savoir s 'ils sont fréquents
- 1000 items => 2^{1000} ensembles à considérer



Vers un algorithme générique



Construction des règles

- Pour chaque ensemble fréquent X, chaque sous-ensemble est choisi comme antécédent de la règle, le reste devenant la partie conséquent.
- Comme X est fréquent, tous les sous-ensembles sont fréquents (Propriété 3) donc leur support est connu. La confiance d'une règle est calculée et une règle est conservée ou pas selon la confiance minimale.
- Amélioration : (Propriété 7) quand une règle échoue, aucun sous ensembles de l'antécédent n'est à considérer.



Bref historique

- Problématique initiée en 1993
- CPU vs. I/O
- De nombreux algorithmes ...

AIS - *R. Agrawal, T. Imielinski and A. Swami* - *ACM SIGMOD 1993*

SETM - *Houtsma and Swami* - *IBM Technical Record*

APRIORI - *R. Agrawal and R. Srikant* - *VLDB 1994*

PARTITION - *A. Sarasere, E. Omiecinsky and S. Navathe* - *VLDB 1995*

SAMPLING - *H. Toivonen* - *VLDB 1996*

DIC - *S. Brin, R. Motwani, J.Ulman and S. Tsur* - *ACM SIGMOD 1997*

PrefixSpan - *J. Pei, J. Han,* - *ICDE'01*

SPADE - *M. Zaki* - *Machine Learning'01*

....2006, ...2010, 2014, 2016



L'algorithme APRIORI

- But : minimiser les candidats
- Principe : générer seulement les candidats pour lesquels tous les sous-ensembles ont été déterminés fréquents
- Génération des candidats réalisée avant et de manière séparée de l'étape de comptage



L'algorithme APRIORI

Input : C_k : itemsets candidats de taille k

Output : L_k : itemsets fréquents de taille k

$L_1 = \{\text{items fréquents}\};$

for ($k = 1; L_k \neq \emptyset; k++$) do

C_{k+1} = candidats générés à partir de L_k ;

Pour chaque transaction t de la base de données, incrémenter le compteur de tous les candidats dans C_{k+1} qui sont contenus dans t

L_{k+1} = candidats dans C_{k+1} avec minSupp

return $\cup_k L_k$;



Détails d'APRIORI

- Comment générer les candidats ?
 - Etape 1: auto-jointure sur L_k
 - Etape 2: élagage
- Comment compter le support des candidats ?



Génération des candidats

- Les items de L_{k-1} sont ordonnés par ordre lexicographique

- Etape 1: auto-jointure sur L_{k-1}

INSERT INTO C_k

SELECT $p.item_1, p.item_2, \dots, p.item_{k-1}, q.item_{k-1}$

FROM $L_{k-1} p, L_{k-1} q$

WHERE $p.item_1=q.item_1, \dots, p.item_{k-2}=q.item_{k-2}, p.item_{k-1} < q.item_{k-1}$

- Etape 2: élagage

For each itemset c in C_k do

For each $(k-1)$ -subsets s of c do if (s is not in L_{k-1}) then delete c from C_k



Génération des candidats : exemple

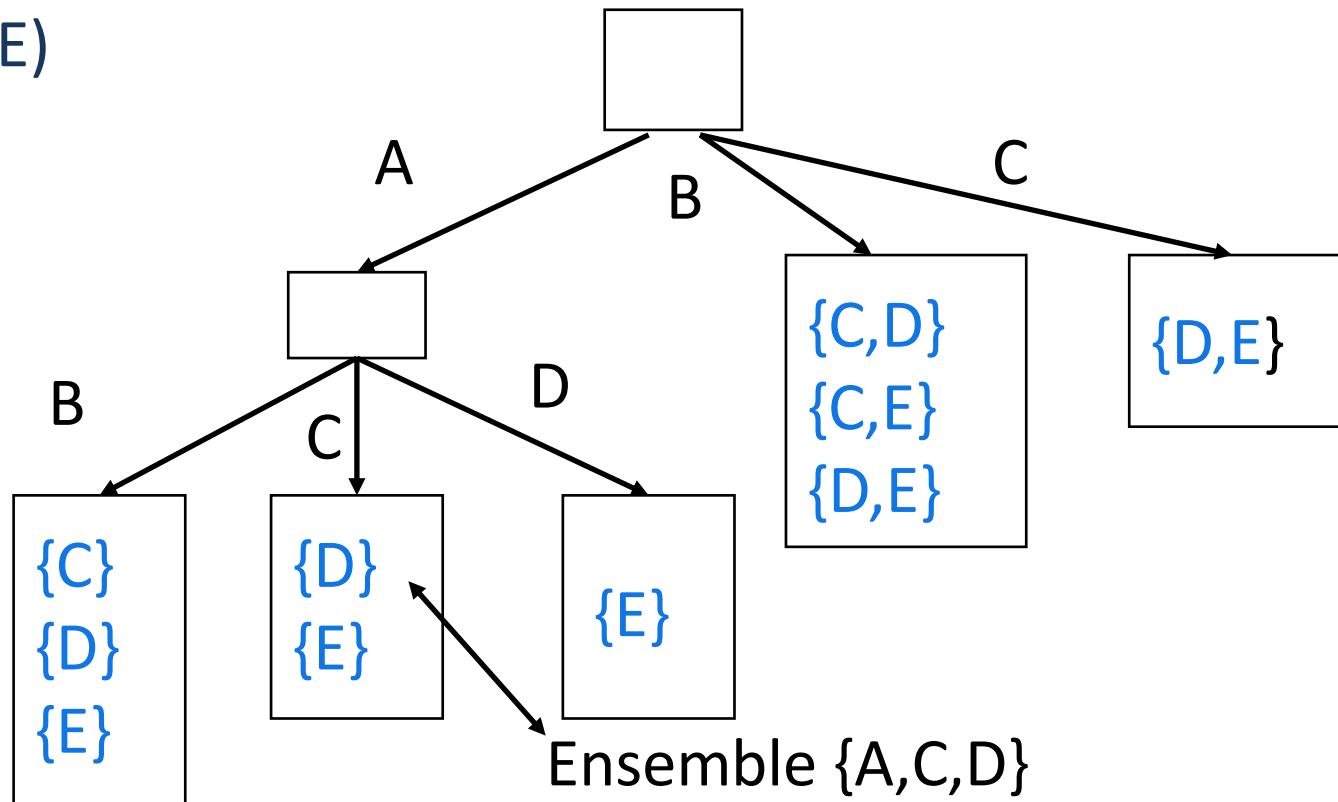
- $L_3 = \{abc, abd, acd, ace, bcd\}$
- Auto-jointure : $L_3 * L_3$
 - $abcd$ à partir de abc et abd
 - $acde$ à partir de acd et ace
- Élagage :
 - $acde$ est supprimé car ade n'est pas dans L_3
- $C_4 = \{abcd\}$



Stockage des candidats

- un arbre (structure de hash-tree)

structure de tous les 3-candidats possibles pour 5 items (A, B, C, D, E)



Comptage du support des candidats

- Parcourir la base. Pour chaque tuple extrait t , compter tous les candidats inclus dedans
 - Rechercher toutes les feuilles qui peuvent contenir les candidats
 - Hachage sur chaque item du tuple et descente dans l'arbre des candidats
- Dans les feuilles de l'arbre vérifier ceux effectivement supportés par t
- Incrémenter leur support



Illustration

CID	Items
1	A B
2	A B C D E F
3	B D G
4	B E G
5	D F G
6	DEG
7	B E
8	B D E F

Support minimal = 1



Illustration

C1	Support
A	2
B	6
C	1
D	5
E	5
F	3
G	4



$L_1 = \{\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{G\}\}$ 1-itemsets fréquents

Illustration

C2	Support	C2	Support
AB	2	CD	1
AC	1	CE	1
AD	1	CF	1
AE	1	CG	0
AF	1	DE	3
AG	0	DF	3
BC	1	DG	3
BD	3	EF	2
BE	4	EG	2
BF	2	FG	1
BG	2		

2-itemsets fréquents $\{\{A,B\}, \{A,C\}, \{A,D\}, \{A,E\}, \{A,F\}, \{B,C\}, \{B,D\}, \{B,E\}, \{B,F\}, \{B,G\}, \{C,D\}, \{C,E\}, \{C,F\}, \{D,E\}, \{D,F\}, \{D,G\}, \{E,F\}, \{E,G\}, \{F,G\}\}$



Illustration

C3	Support	C3	Support
ABC	1	BDE	2
ABD	1	BDF	2
ABE	1	BDG	1
ABF	1	BEF	2
ACD	1	BEG	1
ACE	1	BFG	0
...
BCF	1	EFG	0

$$L_3 = \{\{A, B, C\}, \{A, B, D\}, \{A, B, E\}, \{A, B, F\}, \{A, C, D\}, \dots \{D, F, G\}\}$$

$\{B, C, G\}$ élagué par Apriori-Gen car $\{C, G\}$ n'appartient pas à L_2



Illustration

C4	Support	C4	Support
ABCD	1	ACEF	1
ABCE	1	ADEF	1
ABCF	1	BCDE	1
ABDE	1	BCDF	1
ABDF	1	BCEF	1
ABEF	1	BDEF	2
ACDE	1	BDEG	0
ACDF	1	CDEF	0

$$L_4 = \{\{A, B, C, D\}, \{A, B, C, E\}, \{A, B, C, F\}, \dots \{C, D, E, F\}\}$$

$\{B, D, F, G\}$, $\{B, E, F, G\}$ élagués car $\{B, F, G\}$ n'appartient pas à L_3

$\{D, E, F, G\}$ élagué car $\{E, F, G\}$ n'appartient pas à L_3



Illustration

C6	Support
ABCDEF	1

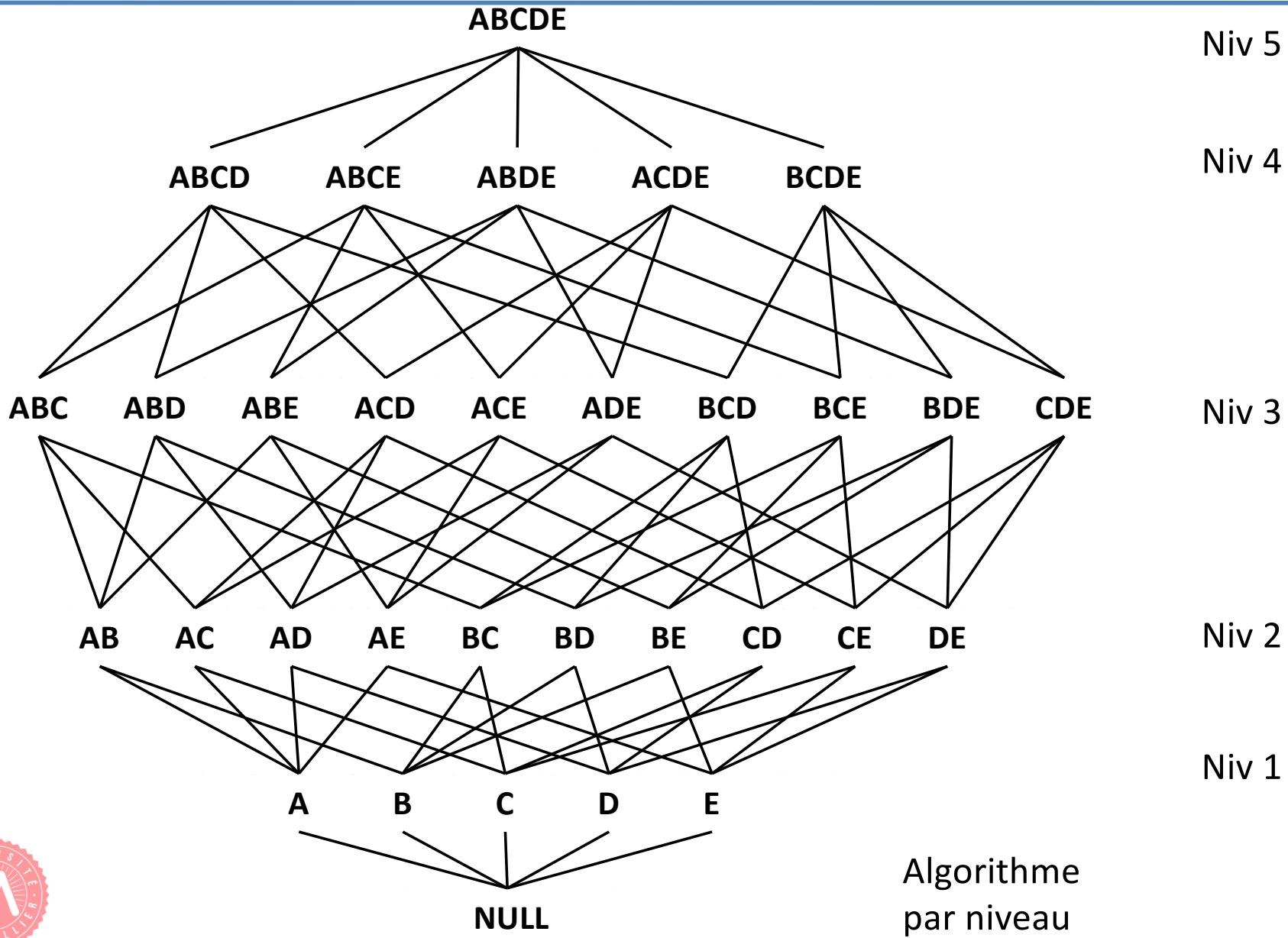
6-itemsets fréquents $L_6 = \{\{A,B,C,D,E,F\}\}$

$C_7 = \{\emptyset\} \Rightarrow$ l'algorithme se termine.

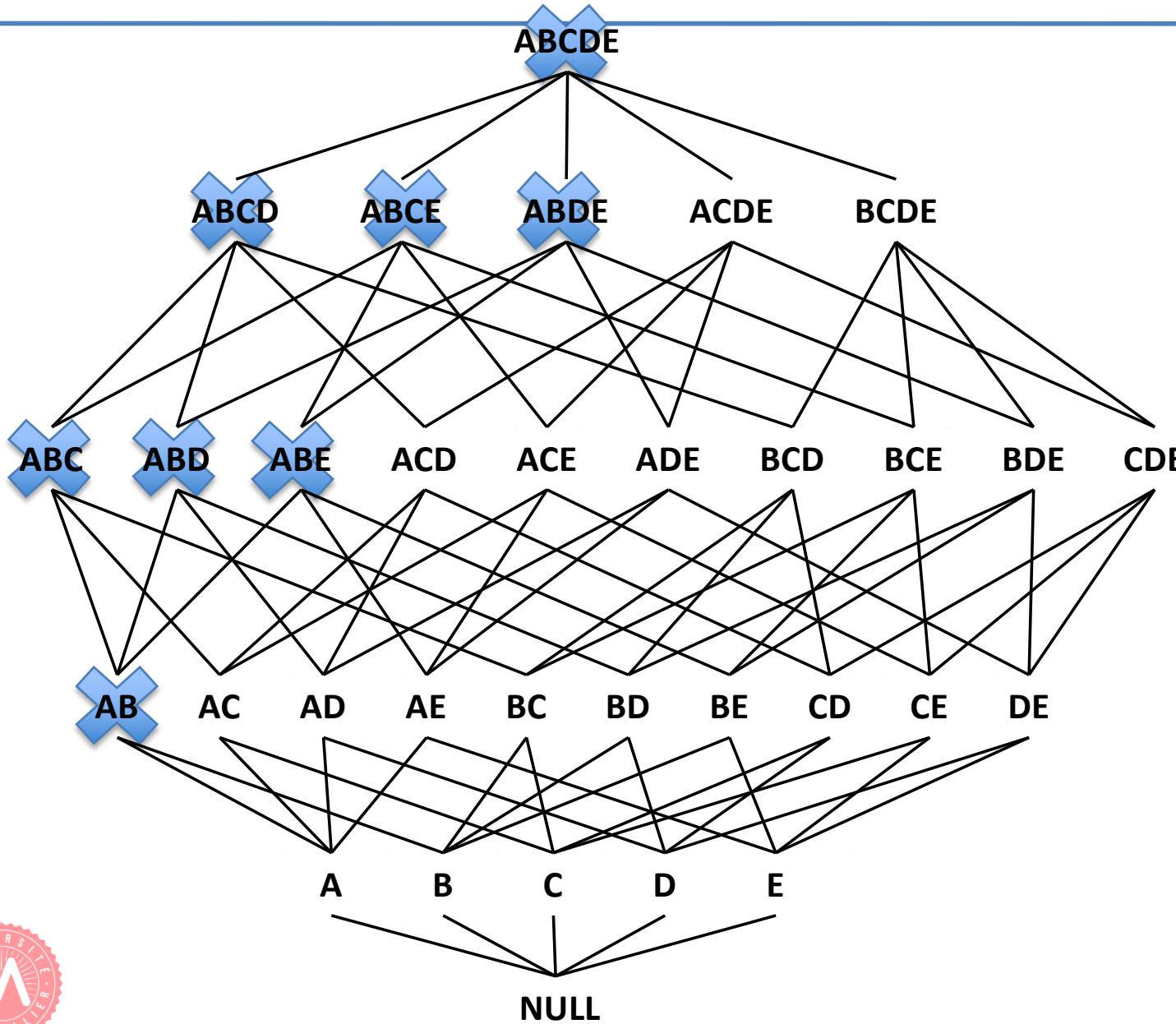
7 balayages pour déterminer tous les itemsets fréquents



Espace de recherche



Espace de recherche



Principe
d'Apriori

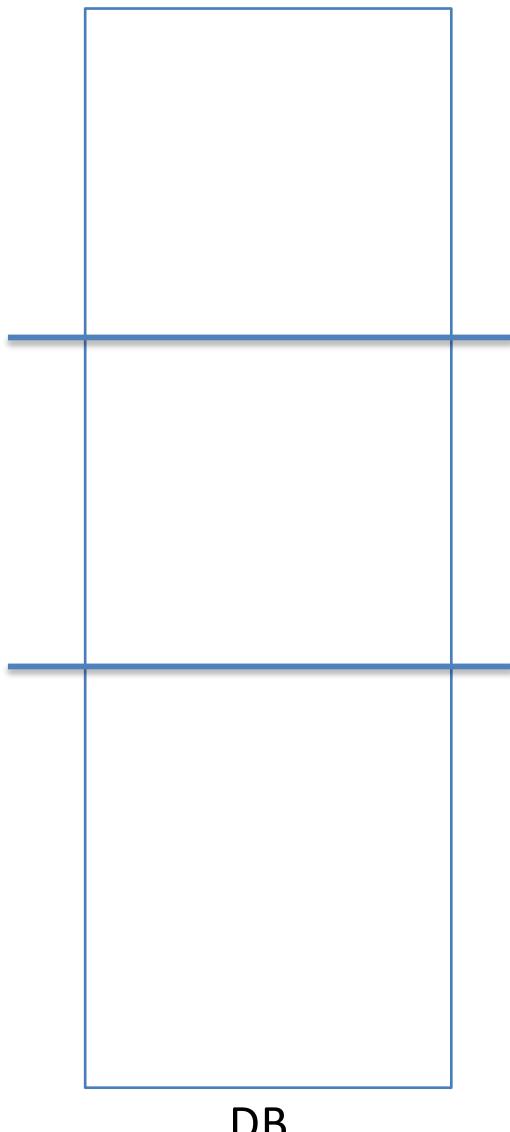
AB n'est pas
fréquent

Partition

- But : Réduire le nombre de passes
- Principe :
 - partitionner la base de manière à ce que chaque partition tienne en mémoire centrale (utilisation d'Apriori pour chaque partition)
 - 2 passes sur la base



Partition



Partition 1
Application d'Apriori en mémoire centrale

Partition 2
Application d'Apriori en mémoire centrale

Partition 3
Application d'Apriori en mémoire centrale



Partition (cont.)

- Phase 1 : Division de la base
 - Traiter les partitions une par une : les itemsets fréquents sont fusionnés pour générer l'ensemble de tous les itemsets fréquents potentiels
- Phase 2 : le support de ces itemsets est calculé

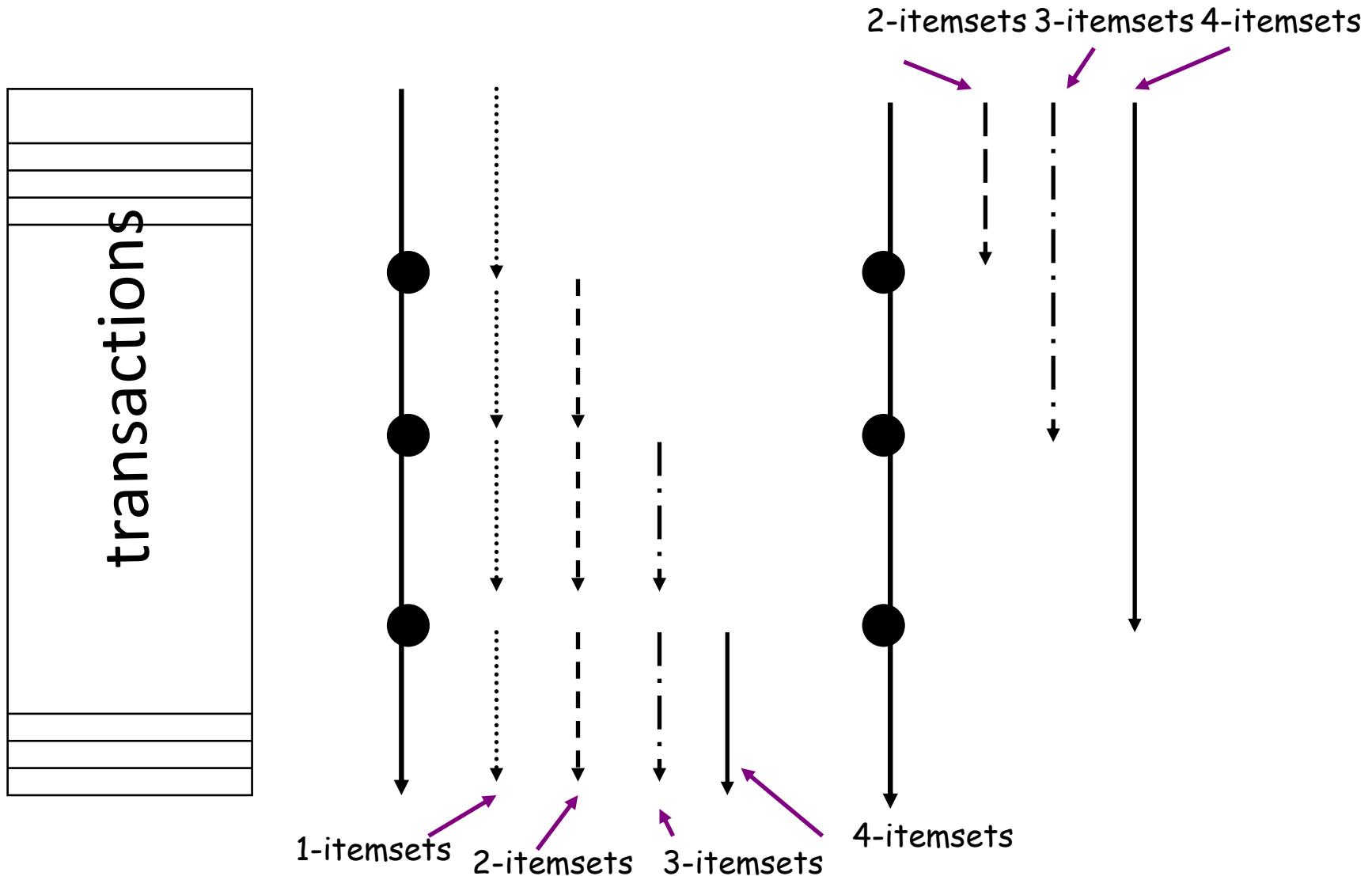


DIC (Dynamic Itemset Counting)

- But : réduction du nombre de balayage de la base
- Lecture par blocs de M transactions
- Essayer de générer le plus vite possible, i.e. à la fin de M , des $(k+1)$ -itemsets pour les rechercher dans les prochaines M transactions



DIC (Cont.)

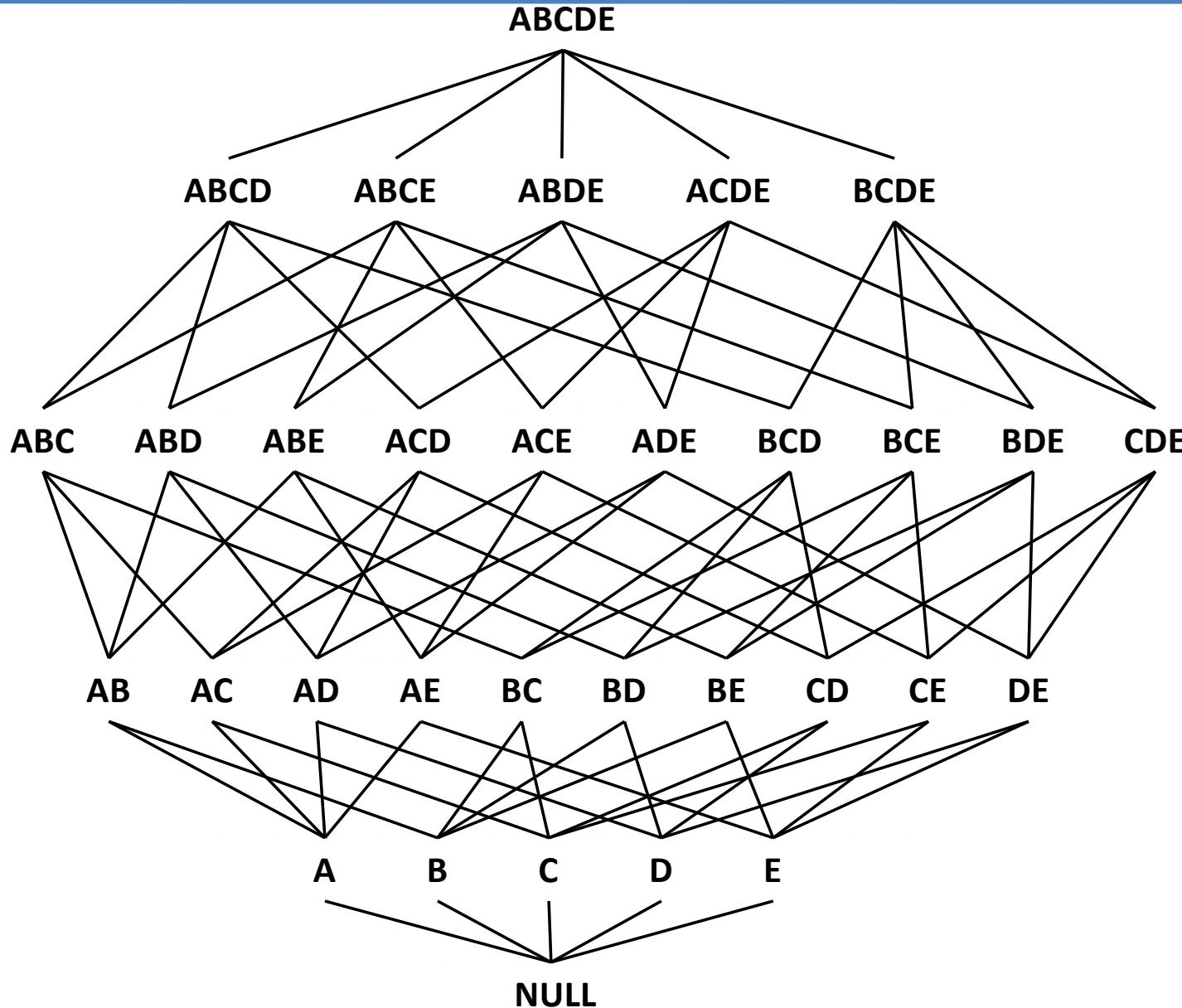


Sampling

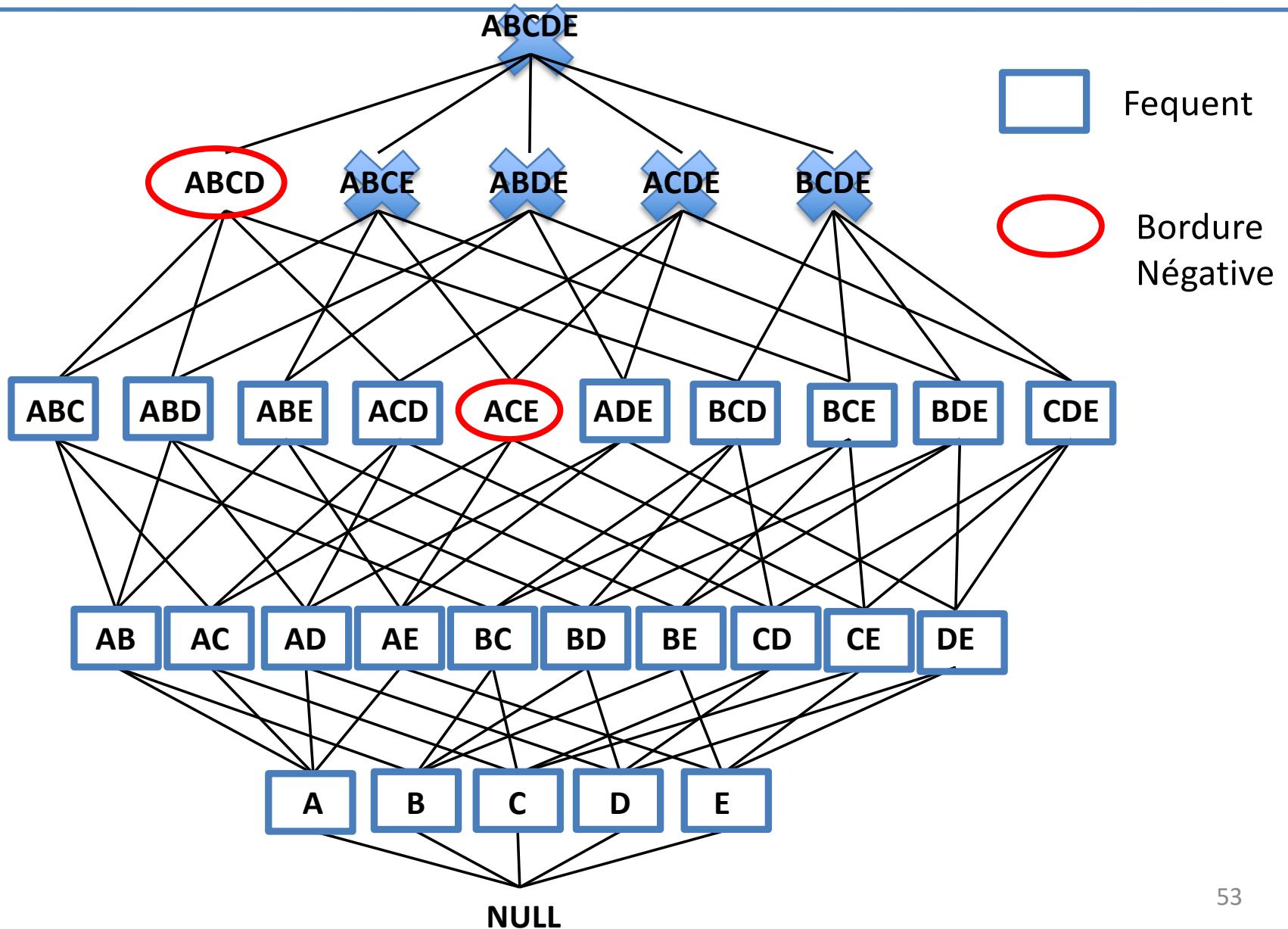
- Idée : prendre un ensemble aléatoire qui réside en mémoire centrale et rechercher tous les itemsets fréquents
- Très efficace : 1 passe, 2 passes au pire
- Basée sur la bordure négative



Bordure Négative



Bordure Négative



Sampling (cont.)

- Algorithme

support minimum, petit support minimum, une base et un échantillon de la base

- 1 - prendre un échantillon de la base
- 2 - Calculer les fréquents avec petit support minimum en mémoire centrale : Fréquents et Bordure
- 3 - Evaluer la fréquence des itemsets fréquents et de la bordure négative sur le reste de la base
- 4 - Retourner le résultat et les éventuels manques



Sampling (cont.)

- $D = 10 \text{ millions de tuples} - A \dots F - \text{support minimum} = 2\% -$
Echantillon s de 20 000 tuples petit support minimum = 1,5%

Pour l'échantillon avec 1,5% : $F=\{\{A,B,C\},\{A,C,F\},\{A,D\},\{B,D\}\}$

Bordure négative = BN= $\{\{B,F\},\{C,D\},\{D,F\},\{E\}\}$

- Evaluer F et BD sur le reste de la base avec 2%
 - 1 - *on trouve $\{A,B\},\{A,C,F\}$ en une passe*
 - 2 - *si $\{B,F\}$ devient fréquent sur D => manque peut être $\{A,B,F\}$*
=> reporter l'erreur et effectuer une seconde passe



MaxMiner : Mining Max-patterns

- But : rechercher les longs itemsets fréquents
- Max-patterns : bordures de motifs fréquents
 - Un sous-ensemble d'un max-pattern est fréquent
 - Un sur-ensemble d'un max-pattern est non fréquent
- Parcours en largeur et en profondeur



MaxMiner : Mining Max-patterns (cont.)

- 1er passage: rechercher les items fréquents
 - A, B, C, D, E
- 2nd passage: rechercher les support pour
 - AB, AC, AD, AE, **ABCDE**
 - BC, BD, BE, **BCDE**
 - CD, CE, **CDE**, DE,
- Comme BCDE est un max-pattern, il n'est pas nécessaire de vérifier BCD, BDE, CDE dans les parcours suivants

Tid	Items
10	A,B,C,D,E
20	B,C,D,E,
30	A,C,D,F

minSupp=2



Génération des candidats

- Depuis 2000 « La base peut tenir en mémoire »
- Constat : génération d'un trop grand nombre de candidats
 - si il y a 10^4 1-itemset => génération de 10^7 candidats 2-itemsets
 - Pour un fréquent de 100, il faut générer plus de 10^{30} candidats au total
- Est-il possible de proposer une méthode qui évite de générer des candidats ?



FP-Tree

- 1 - Parcours de la base pour rechercher les 1-fréquents
- 2 - Tri des fréquents dans l'ordre décroissant

TID	Items	Items triés
1	I1, I2, I5	I2, I1, I5
2	I2, I4	I2, I4
3	I2, I3	I2, I3
4	I1, I2, I4	I2, I1, I4
5	I1, I3	I1, I3
6	I2, I3	I2, I3
7	I1, I3	I1, I3
8	I1, I2, I3, I5	I2, I1, I3, I5
9	I1, I2, I3	I2, I1, I3

$$L = [\quad I2:7, \\ I1:6, \\ I3: 6, \\ I4 : 2, \\ I5 : 2]$$



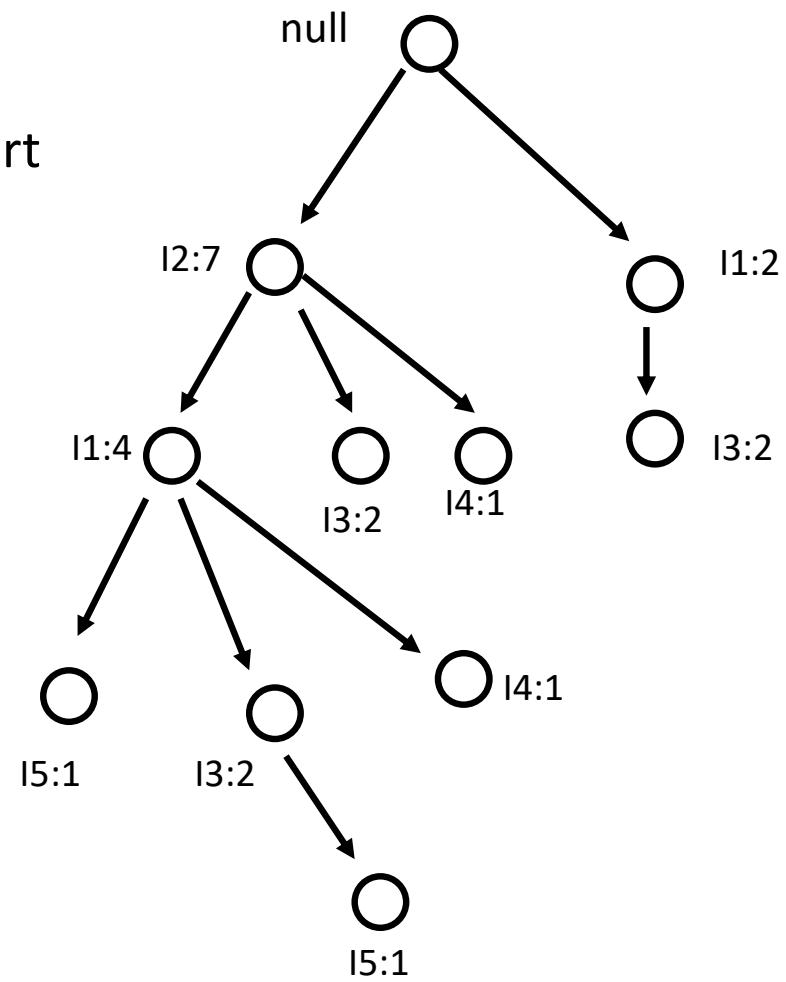
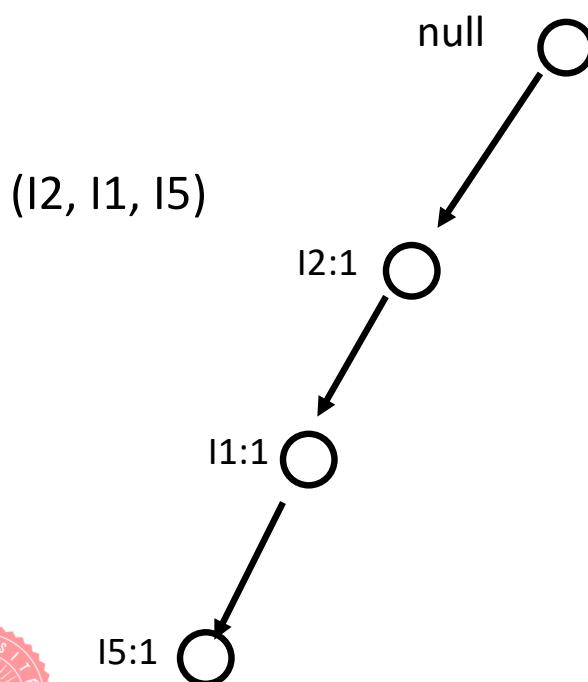
FP-Tree (cont.)

Parcourir les transactions de la base

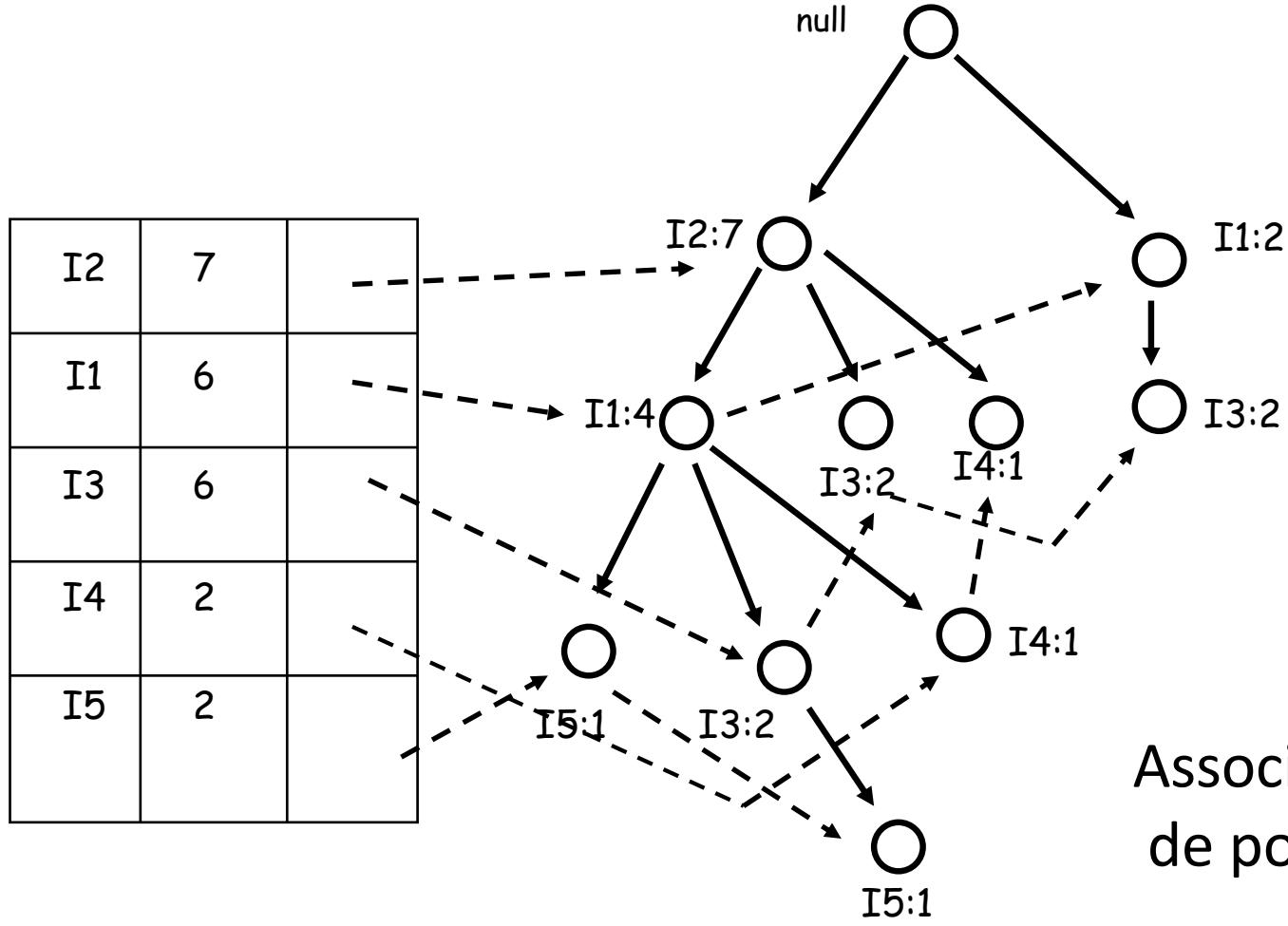
Création du FP-Tree :

« faire glisser les transactions dans l'arbre »

- Une branche existe : incrémenter le support
- Créer la branche autrement

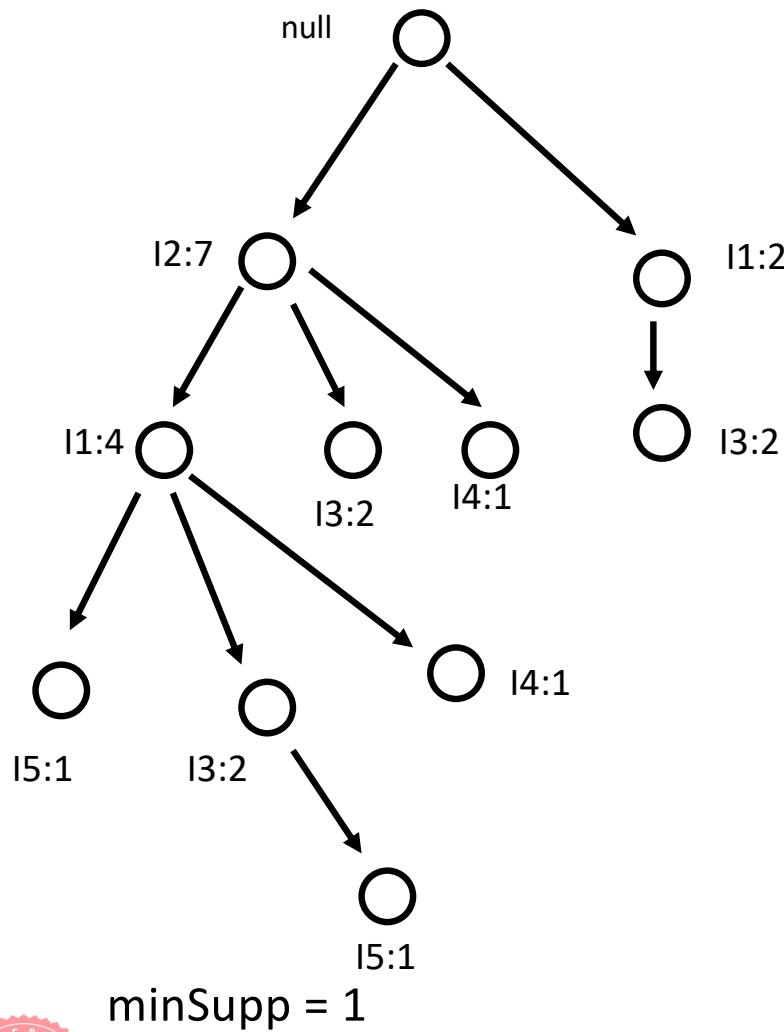


FP-Tree (cont.)



Association d'un tableau
de pointeurs trié

FP-Tree (cont.)



On commence par ceux dont le support est le plus faible

Pour I5

chemins pour I5 <I2 I1 I5:1> et <I2 I1 I3 I5:1>

en considérant I5 comme suffixe on a :

<I2 I1 : 1>

<I2 I1 I3 : 1>

=> <I2 : 2, I1 : 2> (support I3 = 1)

Génération : I2 I5 : 2

I1 I5 : 2

I2 I1 I5 : 2

Pour I4

Avec I4 comme suffixe

<I2 I1 : 1> et <I2 : 1> => fréquent I2 I4 : 2

Pour I3

Avec I3 comme suffixe

<I2 I1 : 2>, <I2 : 2>, <I1 : 2>

=> fréquents : I2 I3 : 4 I1 I3 : 2 I2 I1 I3 : 2

...

Bénéfices de FP-tree

- Préserve l'information complète pour l'extraction d'itemsets
 - Pas de passage supplémentaire sur la base
- Approche Compacte
 - Les items sont triés dans un ordre décroissant de fréquence : plus ils apparaissent fréquemment plus ils seront partagés
 - Ne peut jamais être plus grand que la base d'origine (sans compter les liens, les nœuds et les compteurs)



Trop de fréquents

TID	Items
1	A,B,C,D
2	A,B,C
3	A,B,C
4	B,C,D

Avec support minimal = 2

$$A = 3/4$$

$$B = 4/4$$

$$C = 4/4$$

$$D = 2/4$$

$$AB = 3/4$$

$$AC = 3/4$$

$$BC = 4/4$$

$$BD = 2/4$$

$$ABC = 3/4$$

$$BCD = 2/4$$

4 tuples dans
la base de données

10 itemsets extraits



Comment réduire les fréquents ?

- Recherche des itemsets fréquents maximaux (les plus grands itemsets)
- Recherche des itemsets fermés fréquents
 - itemsets maximaux pour lesquels il n'existe pas de super ensemble avec la même valeur de support
 - Cela consiste à rechercher des classes d'équivalence dans le treillis
- D'autres mesures d'intérêt : dérivables, clos, Jaccard, ...



Cas des données corrélées

- D'autres types d'algorithmes
 - Utilisation du treillis et de ses propriétés
 - Recherche des générateurs
- Close, Close+, Charm ...

$$M = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$



Quelques conclusions

- De nombreux travaux
 - De nouvelles approches condensées
 - De nouvelles contraintes (réduire l'espace de recherche, prise en compte d'expressions régulières)
 - Préservation de la vie privée
- Approches Incrémentales
- Règles plus générales
- Définir de nouvelles mesures (lift, implication, ...)



Règles d'association incrémentales

- Générer les règles dans une base dynamique
- Problème : les algorithmes considèrent des bases statiques
- Objectifs :
 - Chercher les itemsets fréquents dans D
 - Chercher les itemsets fréquents dans $D \cup \{\Delta D\}$
- Doit être fréquent dans D ou ΔD
- Sauvegarder tous les fréquents, la bordure
- ... Data Streams (Flots de Données)



Des règles plus générales

- Les règles négatives
 $\text{Expr}(C_i) \rightarrow \text{Expr}(C_j)$ avec AND, OR, NOT
- Les règles sur plusieurs dimensions
- Les règles à attributs variables
 $\text{Age} \in [x,y] \Rightarrow \text{Salaire} > 45 \text{ K€} (5\%; 30\%)$
- Les règles approximatives
- Les règles avec généralisation
Associée à une taxonomie



D'autres mesures

Articles	A	B	C	A, B	A, C	B, C	A, B, C
Fréquences (%)	45	42,5	40	25	20	15	5

- Si on considère les règles à trois articles, elles ont le même support 5%. Le niveau de confiance est alors :

Règle	Confiance
$A, B \rightarrow C$	0,20
$A, C \rightarrow B$	0,25
$B, C \rightarrow A$	0,33

- La règle « $B, C \rightarrow A$ » possède la plus grande confiance. si B et C apparaissent simultanément dans un achat alors A y apparaît aussi avec une probabilité estimée de 33%.



D'autres mesures (cont.)

Articles	A	B	C	A, B	A, C	B, C	A, B, C
Fréquences (%)	45	42,5	40	25	20	15	5

- A apparaît dans 45% des achats. Il vaut donc mieux prédire A sans autre information que de prédire A lorsque B et C apparaissent.
- Le lift permet de comparer le résultat de la prédiction en utilisant la fréquence du résultat

$$\text{Lift} = \text{confiance} / \text{fréquence(résultat)}$$



D'autres mesures (cont.)

- Une règle est intéressante lorsque le lift est supérieur à 1. Pour les règles choisies, on trouve :

Règle	Confiance	Freq(résultat)	Amélioration
A, B → C	0.20	40%	0.50
A,C → B	0.25	42.5%	0.59
B,C → A	0.33	45%	0.74

- Par contre, pour la règle « $A \rightarrow B$ », B possède un support de 25%, la règle une confiance de 0.55 et un lift de **1.31**, cette règle est donc la meilleure.
- En règle générale, la meilleure règle est celle qui contient le moins d'articles.

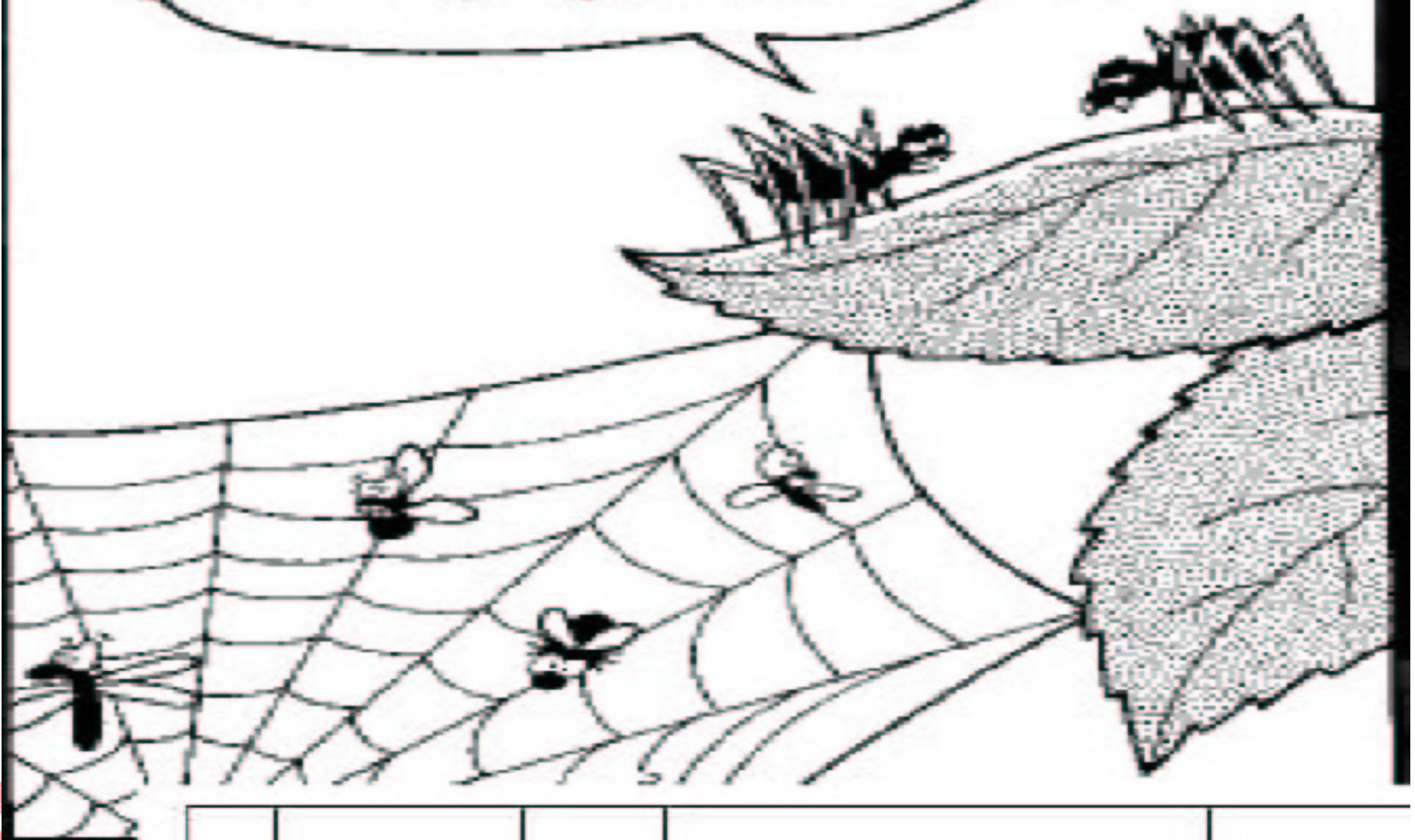


Utilité des règles

- La règle utile contenant des informations de qualité qui peuvent être mises en pratique
ex : le samedi, les clients des épiceries achètent en même temps de la bière et des couches
- Résultats connus par quiconque
ex : les clients des épiceries achètent en même temps du pain et du beurre
- Résultats inexplicables difficiles à situer et donc à expliquer
ex : lors de l'ouverture d'une quincaillerie, parmi les articles les plus vendus on trouve les abattants de toilette



EXCUSE ME A SEC... I WANT
TO CHECK HOW MANY HITS I GOT
ON MY WEBSITE...



Web Usage Mining

- Analyse de l'usage des visiteurs sur un site Web
- Les pages contiennent l'information
- Les liens sont des « routes » (hyperliens)
- Comment les personnes naviguent-elles sur Internet ?
 - Web Usage Mining (Clickstream Analysis)
 - Information sur les chemins de navigation disponibles dans des fichiers logs.
- Principe :
intégrer et « fouiller » ces données pour en produire de l'information et de la connaissance



Web Usage Mining

- Pourquoi analyse l'usage des sites Web ?
- La connaissance sur la manière dont les visiteurs utilisent un site Web permet de :
 - Fournir une aide pour réorganiser site
 - Aider le concepteur à positionner l'information importante que les visiteurs recherchent.
 - Précharger et cacher les pages
 - Fournir des sites adaptatifs (personnalisation)
 - Eviter le « zapping »
- Utile dans le cas du e-commerce



Exemple d'utilisation

Statistiques générales	Performance du site	Retenir les clients
Analyse du contenu	Groupement des clients	Campagne adaptée
Point d'entrée	Ciblages des clients	Campagne ciblée
Parcours	Comportement des clients	Modification dynamique



Web Usage Mining

- De nombreux outils disponibles
- Statistiques générales :
 - Nombre de hits
 - Quelle est la page la plus populaire du site ?
 - Qui a visité le site ?
 - Qu'est ce qui a été téléchargé ?
 - Quels sont les mots clés utilisés pour venir sur le site ?



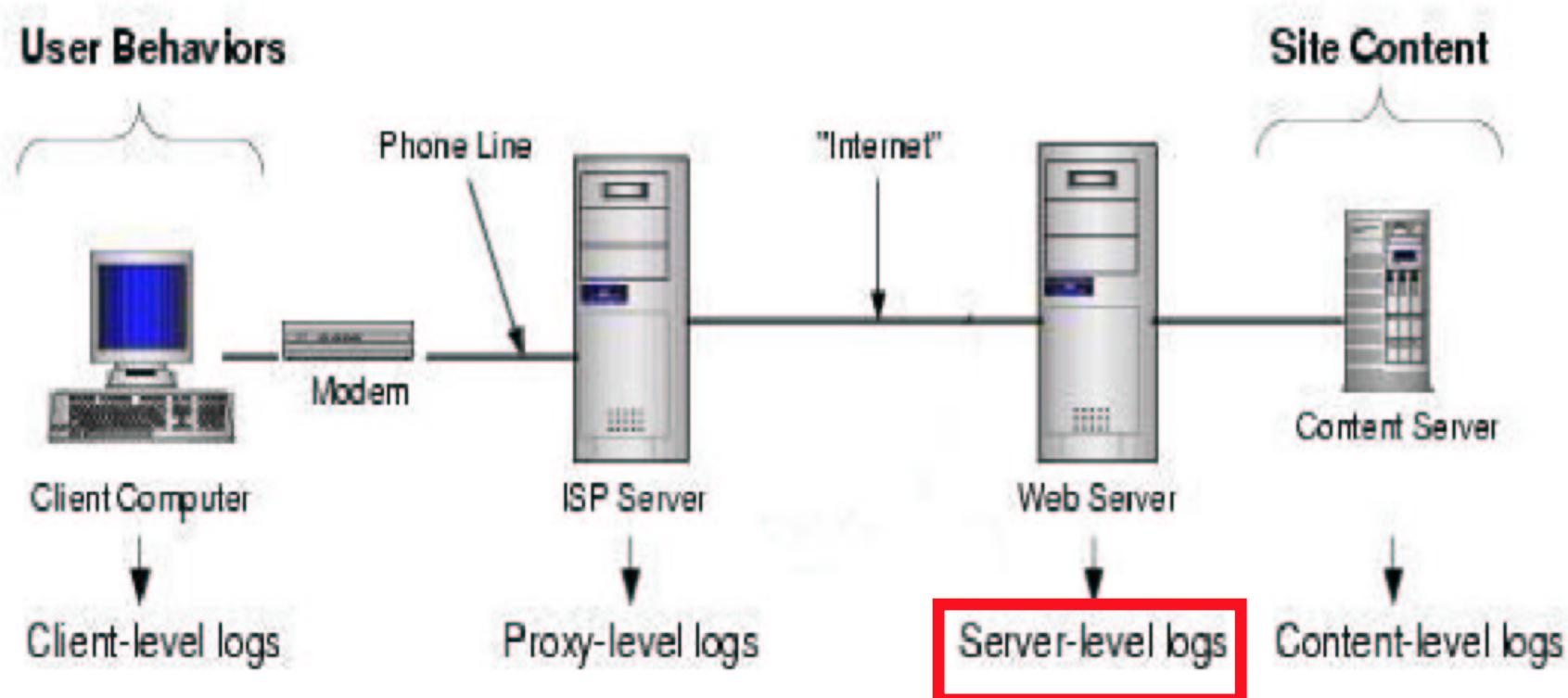
Web Usage Mining



« 75% des **parisiens** qui achètent une **raquette de tennis** achètent **trois mois** après des **chaussures** »
Modification dynamique

Log or Logs?

Information sur les chemins de navigation dans les fichiers logs



Web logs

123.456.78.9 - - [24/Oct/1999:19:13:44 -0400] "GET /Images/tagline.gif HTTP/1.0"

200 1449 <http://www.teced.com/> "Mozilla/4.51 [en] (Win98;I)"

The diagram illustrates the structure of a web log entry. It consists of two lines of text, each with associated labels pointing to specific fields. The first line contains the IP or domain name (123.456.78.9), User Id ([24/Oct/1999:19:13:44 -0400]), Date and Time, and the Request (GET /Images/tagline.gif HTTP/1.0). The second line contains the Status (200), File Size (1449), Referrer URL (<http://www.teced.com/>), Browser (Mozilla/4.51 [en] (Win98;I)), and Cookies.

IP or domain name	User Id	Date and Time	Request
123.456.78.9	[24/Oct/1999:19:13:44 -0400]		"GET /Images/tagline.gif HTTP/1.0"

Status	File Size	Referrer URL	Browser	Cookies
200	1449	http://www.teced.com/	Mozilla/4.51 [en] (Win98;I)	



Web logs

IP Address	Time	Method/URL/Protocol	Sta	Size	Referre d	Agent
123.456.78.9	[25/Apr/1998:03:04:41 –0500	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:34 –0500	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:39 –0500	GET L.html HTTP/1.0	200	4130	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:02 –0500	GET F.html HTTP/1.0	200	5096	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:58 –0500	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:42 –0500	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:55 –0500	GET R.html HTTP/1.0	200	8140	L.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:09:50 –0500	GET C.html HTTP/1.0	200	1820	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:10:02 –0500	GET O.html HTTP/1.0	200	2270	F.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:10:45 –0500	GET J.html HTTP/1.0	200	9430	C.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:12:23 –0500	GET G.html HTTP/1.0	200	7220	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:05:05:22 –0500	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)

Web logs

IP Address	Time	Method/URL/Protocol	Sta	Size	Referer d	Agent
123.456.78.9	[25/Apr/1998:03:04:41 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:34 –0500]	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:39 –0500]	GET L.html HTTP/1.0	200	4130	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:02 –0500]	GET F.html HTTP/1.0	200	5096	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:58 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:42 –0500]	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:55 –0500]	GET R.html HTTP/1.0	200	8140	L.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:09:50 –0500]	GET C.html HTTP/1.0	200	1820	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:10:02 –0500]	GET O.html HTTP/1.0	200	2270	F.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:10:45 –0500]	GET J.html HTTP/1.0	200	9430	C.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:12:23 –0500]	GET G.html HTTP/1.0	200	7220	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:05:05:22 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)

Web logs

IP Address	Time	Method/URL/Protocol	Sta	Size	Referre d	Agent
123.456.78.9	[25/Apr/1998:03:04:41 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:34 –0500]	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:05:39 –0500]	GET L.html HTTP/1.0	200	4130	-	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:02 –0500]	GET F.html HTTP/1.0	200	5096	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:06:58 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:42 –0500]	GET B.html HTTP/1.0	200	2050	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:07:55 –0500]	GET R.html HTTP/1.0	200	8140	L.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:09:50 –0500]	GET C.html HTTP/1.0	200	1820	A.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:10:02 –0500]	GET O.html HTTP/1.0	200	2270	F.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:03:10:45 –0500]	GET J.html HTTP/1.0	200	9430	C.html	Mozilla/3.01 (X11, I, IRIX6.2, IP22)
123.456.78.9	[25/Apr/1998:03:12:23 –0500]	GET G.html HTTP/1.0	200	7220	B.html	Mozilla/3.01 (Win95, I)
123.456.78.9	[25/Apr/1998:05:05:22 –0500]	GET A.html HTTP/1.0	200	3290	-	Mozilla/3.01 (Win95, I)

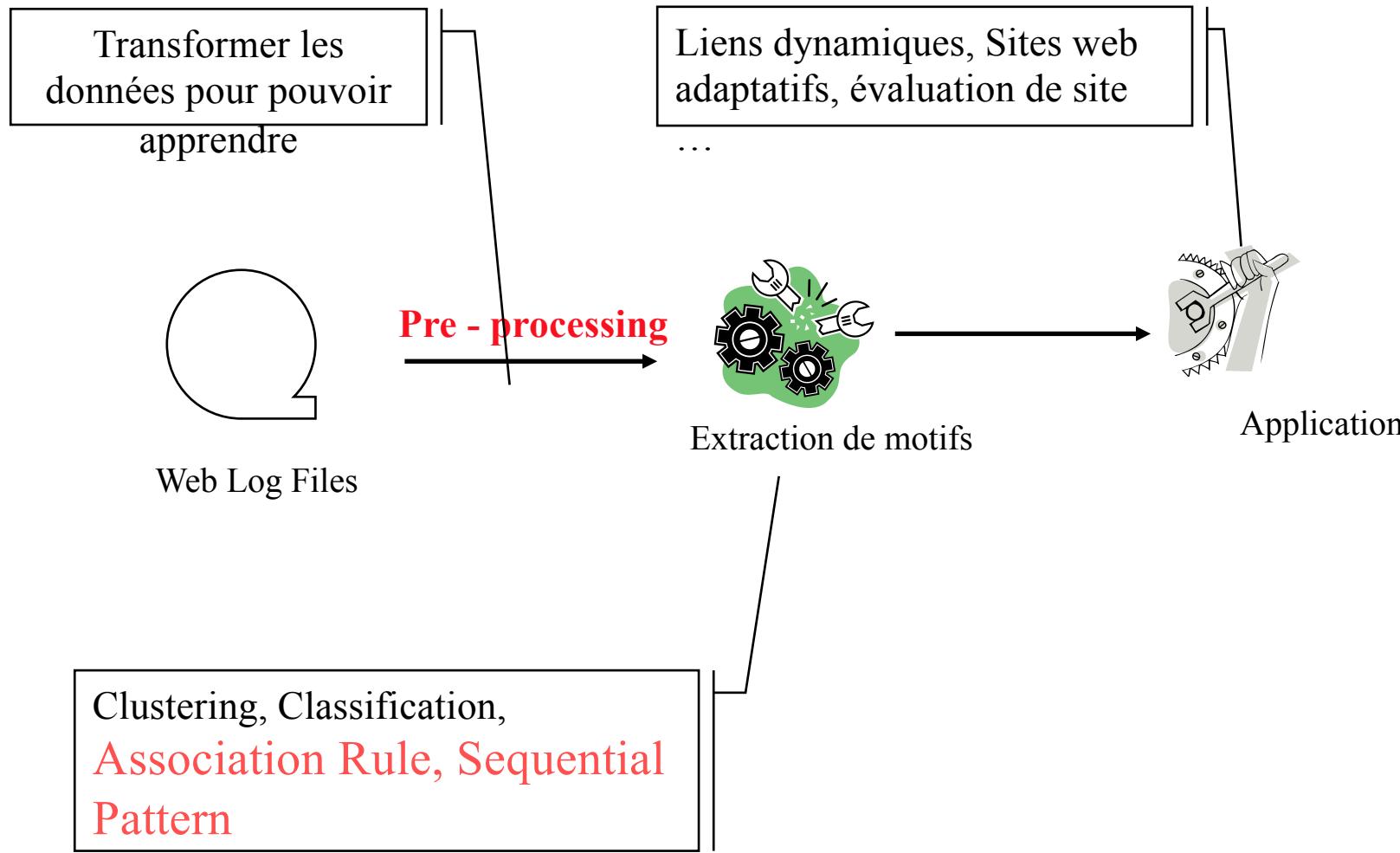


Clients

Dates

Items

KDD pour WUM ?



Pre-traitements

- **Data Filtering - Data Cleaning**
 - Status Code (1xx: Informational, 2xx: Success, 3xx: Redirection, 4xx: Client Error, 5xx: Server Error)
 - Requêtes automatiques (bots, performance monitoring systems)
 - Suppression des entrées concernant des requêtes pour des fichiers graphiques, des frames ...
 - Suppression des entrées générées par des spiders/crawlers (utilisés par les moteurs de recherche)



Web Usage Mining

- Préparation des données (suffixe, éliminations des robots – agents de moteurs)
- Identification de l'utilisateur

Tout n'est pas dans le fichier Access Log

Utilisation d'heuristiques :

Si une page est demandée et qu'elle n'est pas directement liée aux autres pages, il est probable qu'il existe différents utilisateurs sur la même machine

Utilisation des informations sur l'IP, le nom de la machine, le navigateur, des informations temporelles ...



Web Usage Mining

- Problèmes :
 - ID utilisateurs supprimées pour des raisons de sécurité
 - IP individuelles cachées par les proxys
 - Les caches des proxy et du côté clients
- Solutions actuelles :
 - Enregistrement de l'utilisateur – pratique ??
 - Cookies – difficile ??
 - « Cache busting » - augmente le trafic sur le réseau (inutile avec certains proxy)



Web Usage Mining

- Sessions : Comment identifier/définir une transaction d'un visiteur ?
- « Time Oriented »
 - Durée totale d'une session : ≤ 30 minutes
 - Par temps passé sur une page : ≤ 10 minutes/page
- « Navigation Oriented »
 - Le « referrer » est la page précédente, ou le « referrer » n'est pas défini mais demandé dans les 10 secondes, ou le lien de la page précédente à la page courante dans le site web



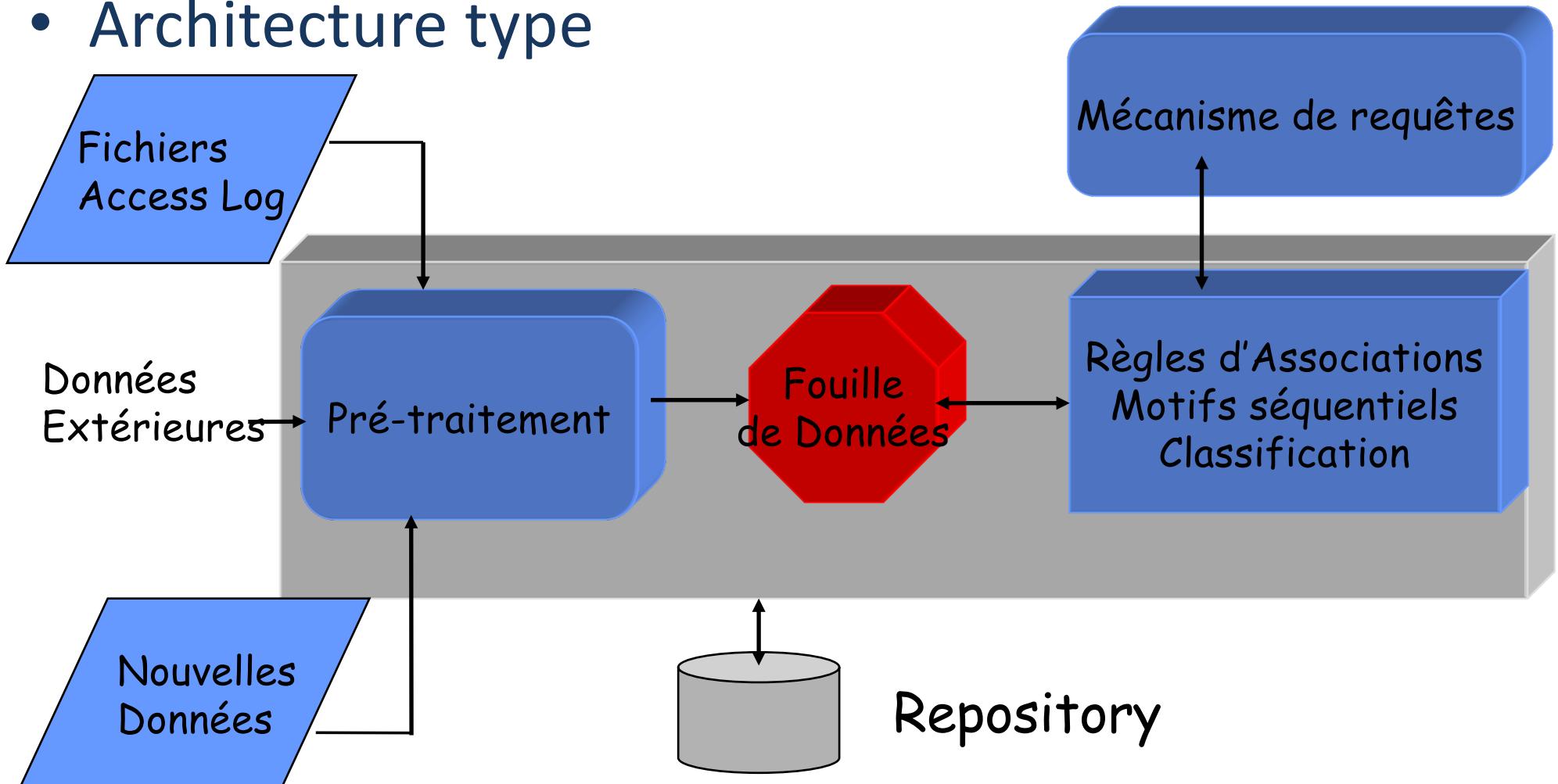
Web Usage Mining

- Sources de données
 - Utilisation de fichiers logs
 - Mais aussi cookies, bases de données des clients,
 -

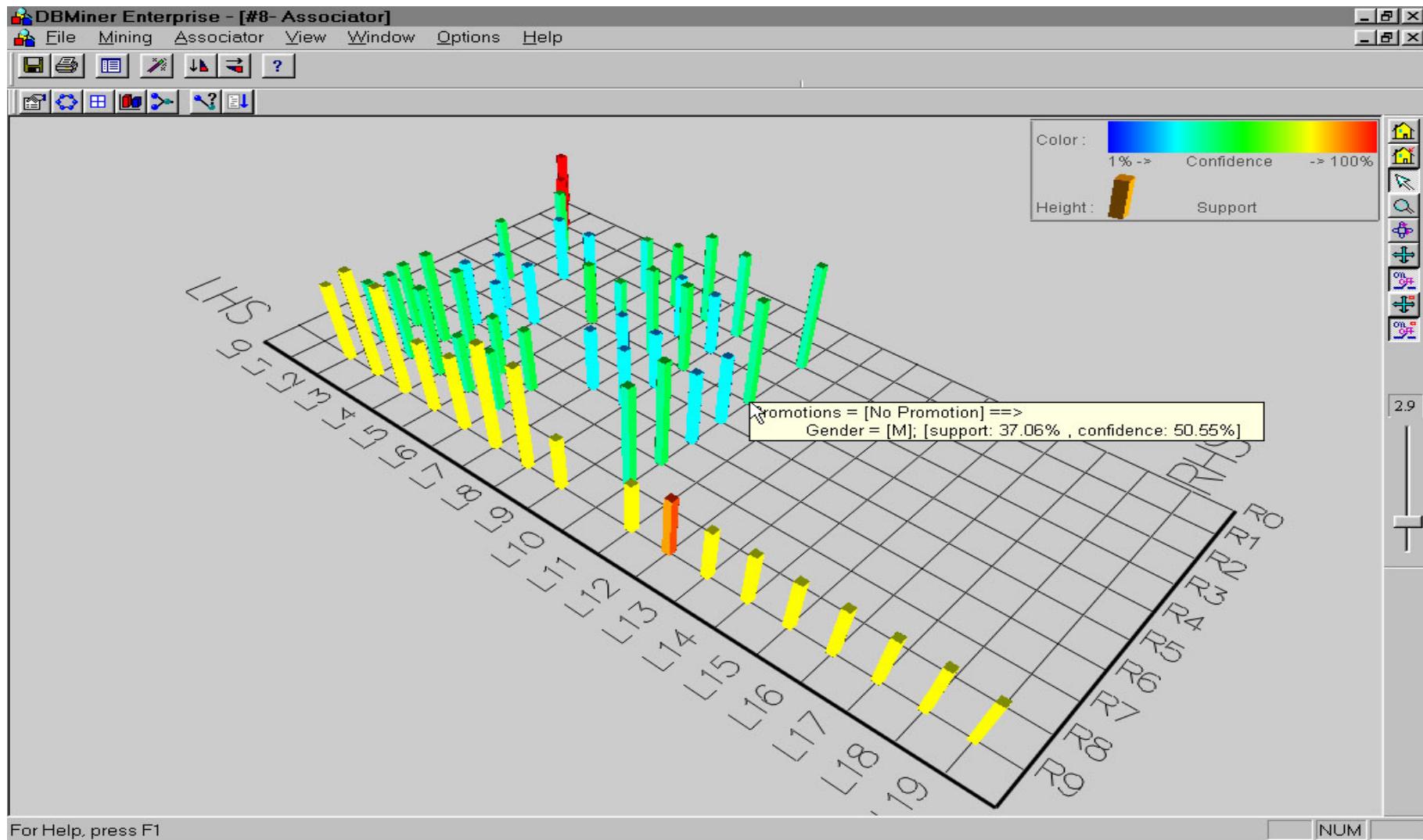


Web Usage Mining

- Architecture type

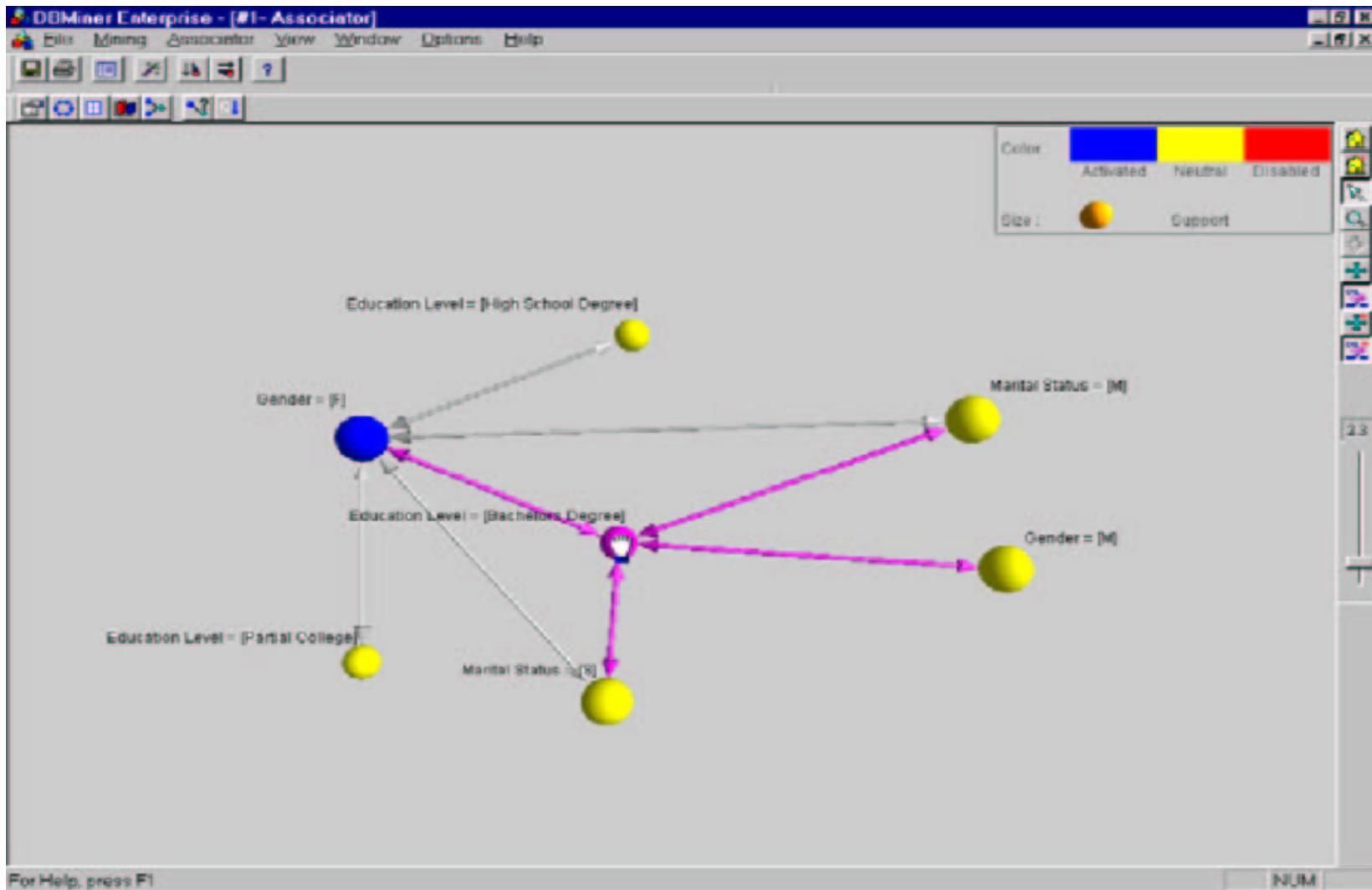


Visualisation

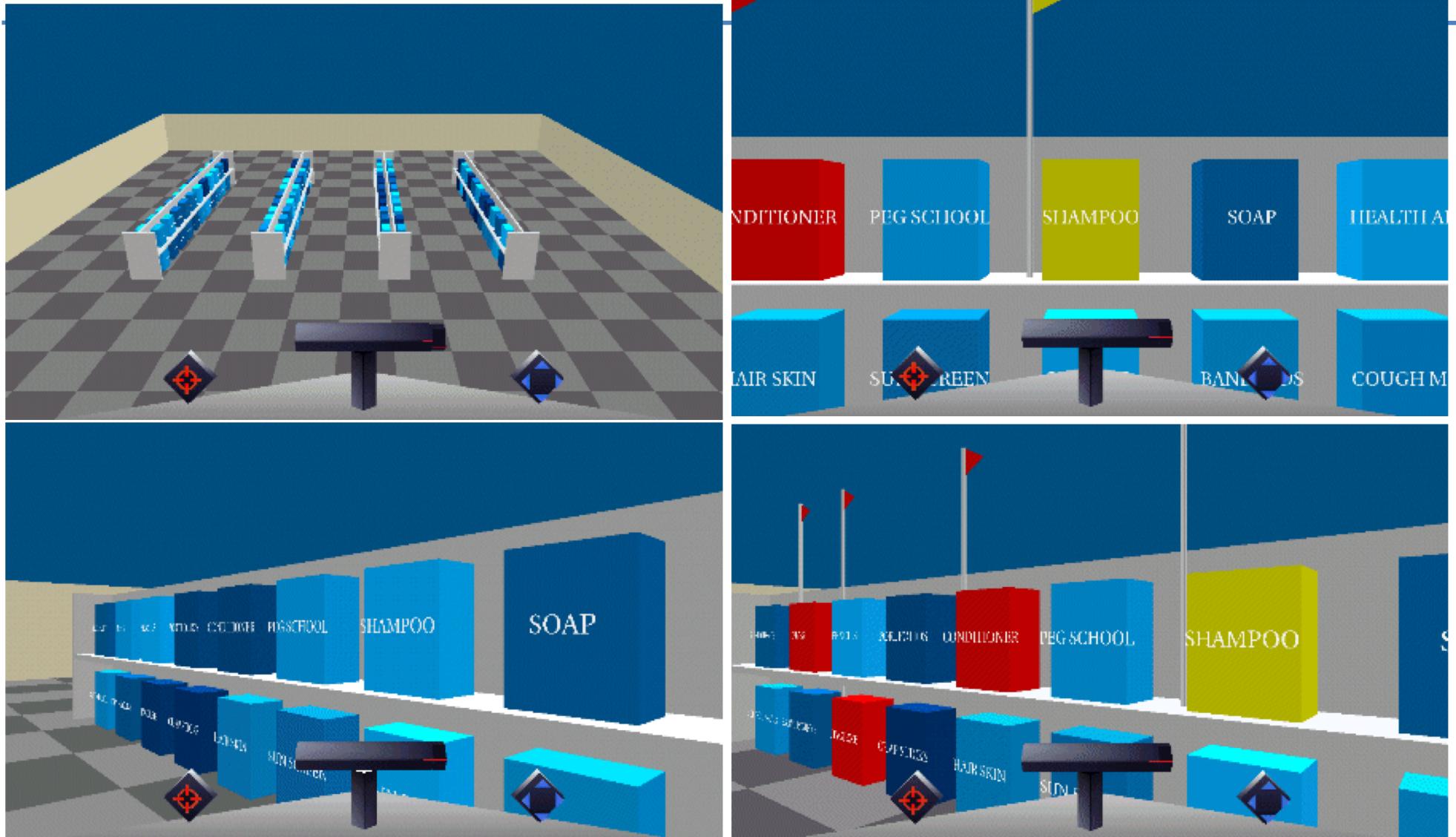


DBMiner (www.dbminer.com)

Visualisation



Visualisation



-
- Des questions ?

