732A61/TDDD41 Data Mining - Clustering and Association Analysis

Lecture 8: Constrained Frequent Itemset Mining

Jose M. Peña
IDA, Linköping University, Sweden

Outline

- Content
  - Monotone and antimonotone constraints
  - Apriori algorithm + (anti) monotone constraints
  - FP grow algorithm + (anti) monotone constraints
  - Convertible (anti) monotone constraints
  - Summary
- Literature
  - Course book. Second edition: 5.5. Third edition: 7.3.
  - Pei, J. and Han, J. Can We Push More Constraints into Frequent Pattern Mining ? In Proc. of the 2000 Int. Conf. on Knowledge Discovery and Data Mining, 2000.

Monotone and Antimonotone Constraints

- A constraint is a function that returns true or false for any itemset, meaning whether the itemset satisfies the constraint or not. For instance:
  - The itemset has support equal or greater than a given value.
  - The sum of the prices of the items in the itemset is greater than 5 units.
  - The most expensive item in the itemset costs less than 5 units.
  - The itemset contains the item "bread".
  - The itemset contains exactly three items.
- A constraint C is monotone when for any itemset A we have that if $C(A) = \mathit{true}$ then $C(B) = \mathit{true}$ where B is any itemset such that $A \subseteq B$. For instance:
  - The itemset contains the item "bread".
- A constraint C is antimonotone when for any itemset A we have that if $C(A) = \mathit{true}$ then $C(B) = \mathit{true}$ where B is any itemset such that $B \subseteq A$. For instance:
  - The support of the itemset is equal or greater than a given value.
- Alternatively, if $C(B) = \mathit{false}$ for some itemset B such that $B \subseteq A$ then $C(A) = \mathit{false}$.
- Note that the apriori property applies to any antimonotone constraint, i.e. we do not need to check the constraint for subsets of A if $C(A) = \mathit{true}$.

Monotone and Antimonotone Constraints

- Examples of monotone constrains:
  - $sum(S) \geq v$ where S is the set of prices of the items in the itemset, and v is a given value.
  - Really monotone ? Only if the prices of all the items are non-negative.
  - $min(S) \leq v$
  - $range(S) \geq v$
- Examples of antimonotone constrains:
  - $sum(S) \leq v$ when the prices of all the items are non-negative.
  - $max(S) \leq v$
  - $range(S) \leq v$

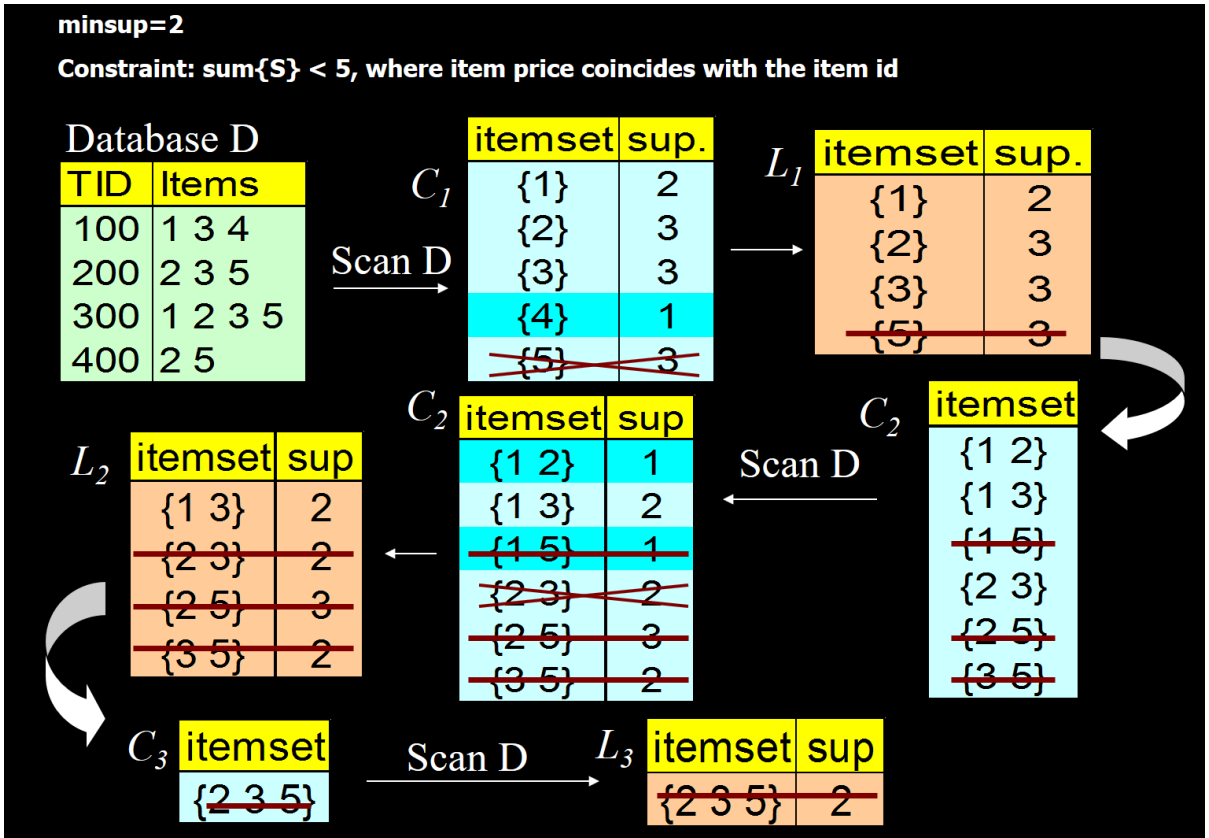
Monotone and Antimonotone Constraints

Constraint	Antimonotone	Monotone
$v \in S$	no	yes
$S \supseteq V$	no	yes
$S \subseteq V$	yes	no
$\min(S) \leq v$	no	yes
$\min(S) \geq v$	yes	no
$\max(S) \leq v$	yes	no
$\max(S) \geq v$	no	yes
$\text{count}(S) \leq v$	yes	no
$\text{count}(S) \geq v$	no	yes
$\text{sum}(S) \leq v$ ($a \in S, a \geq 0$)	yes	no
$\text{sum}(S) \geq v$ ($a \in S, a \geq 0$)	no	yes
$\text{range}(S) \leq v$	yes	no
$\text{range}(S) \geq v$	no	yes
$\text{avg}(S)\ \theta\ v, \theta \in \{=, \leq, \geq\}$	No but convertible	No but convertible
$\text{support}(S) \geq \xi$	yes	no
$\text{support}(S) \leq \xi$	no	yes

Apriori Algorithm + Antimonotone Constraint

Algorithm: apriori(D, *minsup*, C)
Input: A transactional database D, *minsup*, **and an antimonotone constraint** C.
Output: All the large itemsets in D **that satisfy** C.

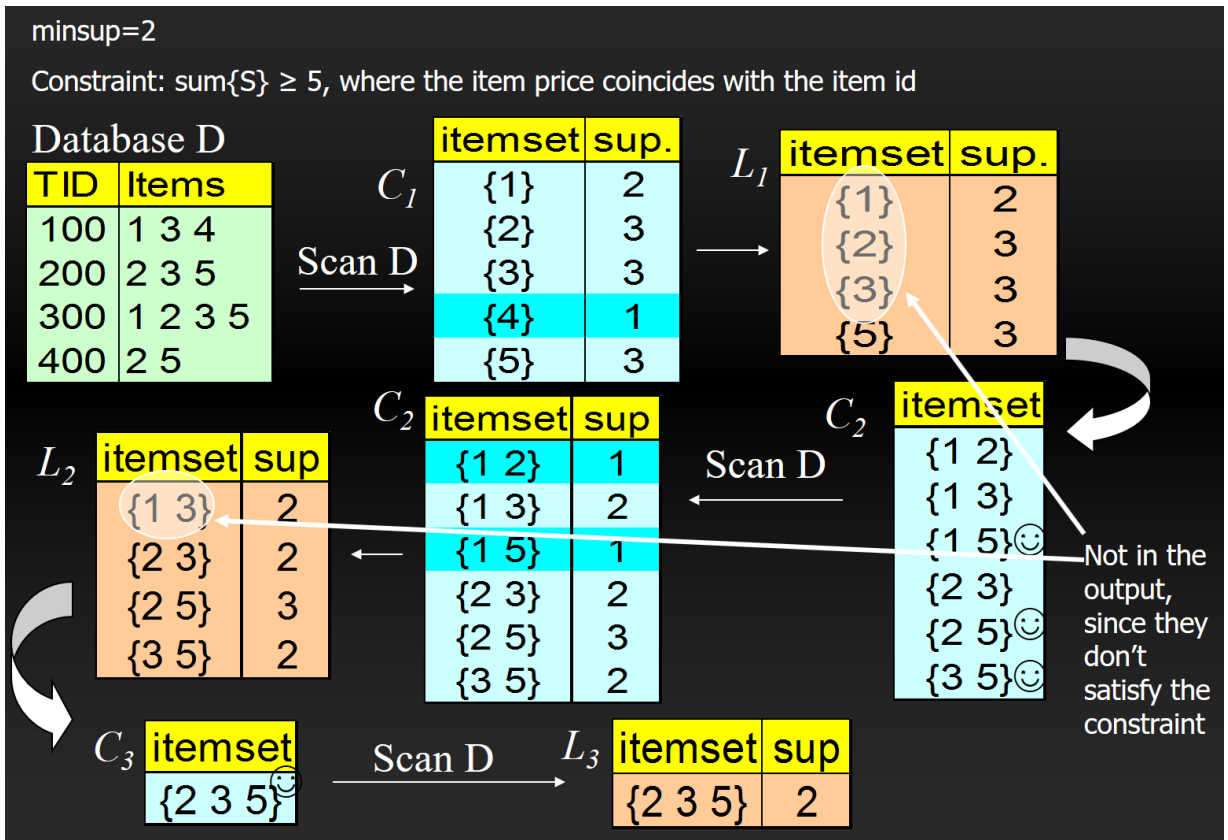
1 $L_1 = \{$ large 1-itemsets **that satisfy** C $\}$
2 for $(k = 2; L_{k-1} \neq \emptyset; k++)$ do
3 $C_k =$ apriori-gen(L_{k-1}) // Generate candidate large k-itemsets
4 for all $t \in D$ do
5 for all $c \in C_k$ such that $c \in t$ do
6 $c.count++$
7 $L_k = \{c \in C_k | c.count \geq minsup$ **and** $C(c) =$ ***true***$\}$
8 return $\bigcup_k L_k$



Apriori Algorithm + Monotone Constraint

Algorithm: apriori(D, *minsup*, $\boldsymbol{C}$)
Input: A transactional database D, *minsup*, **and a monotone constraint** $\boldsymbol{C}$.
Output: All the large itemsets in D **that satisfy** $\boldsymbol{C}$.

```
1  L_1 = { large 1-itemsets }
2  for (k = 2; L_{k-1} ≠ ∅; k++) do
3      C_k = apriori-gen(L_{k-1})      // Generate candidate large k-itemsets
4      for all t ∈ D do
5          for all c ∈ C_k such that c ∈ t do
6              c.count++
7      L_k = {c ∈ C_k | c.count ≥ minsup}
8  return {c ∈ ⋃_k L_k | C(c) = true or C(d) = true for some d ⊆ c}
```

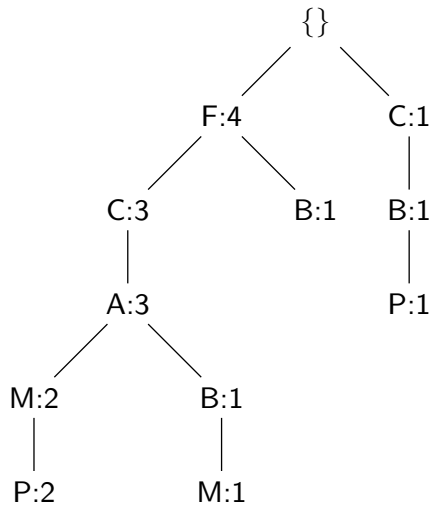


FP Grow Algorithm + Monotone Constraint

Algorithm: FP-grow(*Tree*, α, *minsup*, ***C***)
Input: A FP tree *Tree*, an itemset α, *minsup*, **and a monotone constraint *C***.
Output: All the itemsets in *Tree* that end with α, have *minsup* **and satisfy *C***.

1 **if *C*(α) = *true* then**
2 **replace *C* with C_{true}** **// C_{true} is a constraint that always returns true**
3 for each item X in *Tree* do
4 output the itemset $\beta = X \cup \alpha$ with support=$X.count$ **if *C*(β) = *true***
5 build the β conditional database and the corresponding FP tree $Tree_\beta$
6 if $Tree_\beta$ is not empty then call FP-grow($Tree_\beta$, β, *minsup*, ***C***)

Transaction id	Items bought
1	F, C, A, M, P
2	F, C, A, B, M
3	F, B
4	C, B, P
5	F, C, A, M, P



Item	Conditional database
F	-
C	F:3
A	FC:3
B	FCA:1, F:1, C:1
M	FCA:2, FCAB:1
P	FCAM:2, CB:1

FP Grow Algorithm + Antimonotone Constraint

Algorithm: FP-tree(D, *minsup*, $\boldsymbol{C}$)
Input: A transactional database D, *minsup*, **and an antimonotone constraint $\boldsymbol{C}$**.
Output: The FP tree for D, *minsup* **and $\boldsymbol{C}$**.

1 Count support for each item in D
2 Remove the infrequent items from the transactions in D
3 **Remove the items that do not satisfy $\boldsymbol{C}$ from the transactions in $\boldsymbol{D}$**
4 Sort the items in each transaction in D in support descending order
5 Create a FP tree with a single node T with $T.name = NULL$
6 for each transaction $I \in D$ do
7 insert-tree(I, T)

FP Grow Algorithm + Antimonotone Constraint

Algorithm: FP-grow(*Tree*, α, *minsup*, ***C***)
Input: A FP tree *Tree*, an itemset α, *minsup*, **and an antimonotone constraint *C***.
Output: All the itemsets in *Tree* that end with α, have *minsup* **and satisfy *C***.

1 **let δ denote all the items in *Tree***
2 **if $C(\alpha \cup \delta)$ = *true* then**
3 **replace *C* with C_{true}** **// C_{true} is a constraint that always returns true**
4 for each item X in *Tree* do
5 **if $C(X \cup \alpha)$ = *true* then**
6 output the itemset $\beta = X \cup \alpha$ with support=$X.count$
7 build the β conditional database and the corresponding FP tree $Tree_\beta$
8 if $Tree_\beta$ is not empty then call FP-grow($Tree_\beta$, β, *minsup*, ***C***)

Item	Conditional database
F	-
C	F:3
A	FC:3
B	FCA:1, F:1, C:1
M	FCA:2, FCAB:1
P	FCAM:2, CB:1

M-conditional database

Tid	Items bought
1	F, C, A
2	F, C, A
3	F, C, A, B

After sorting

Tid	Items bought
1	F, C, A
2	F, C, A
3	F, C, A

Output FM and re-start the process for the FM-conditional database ?

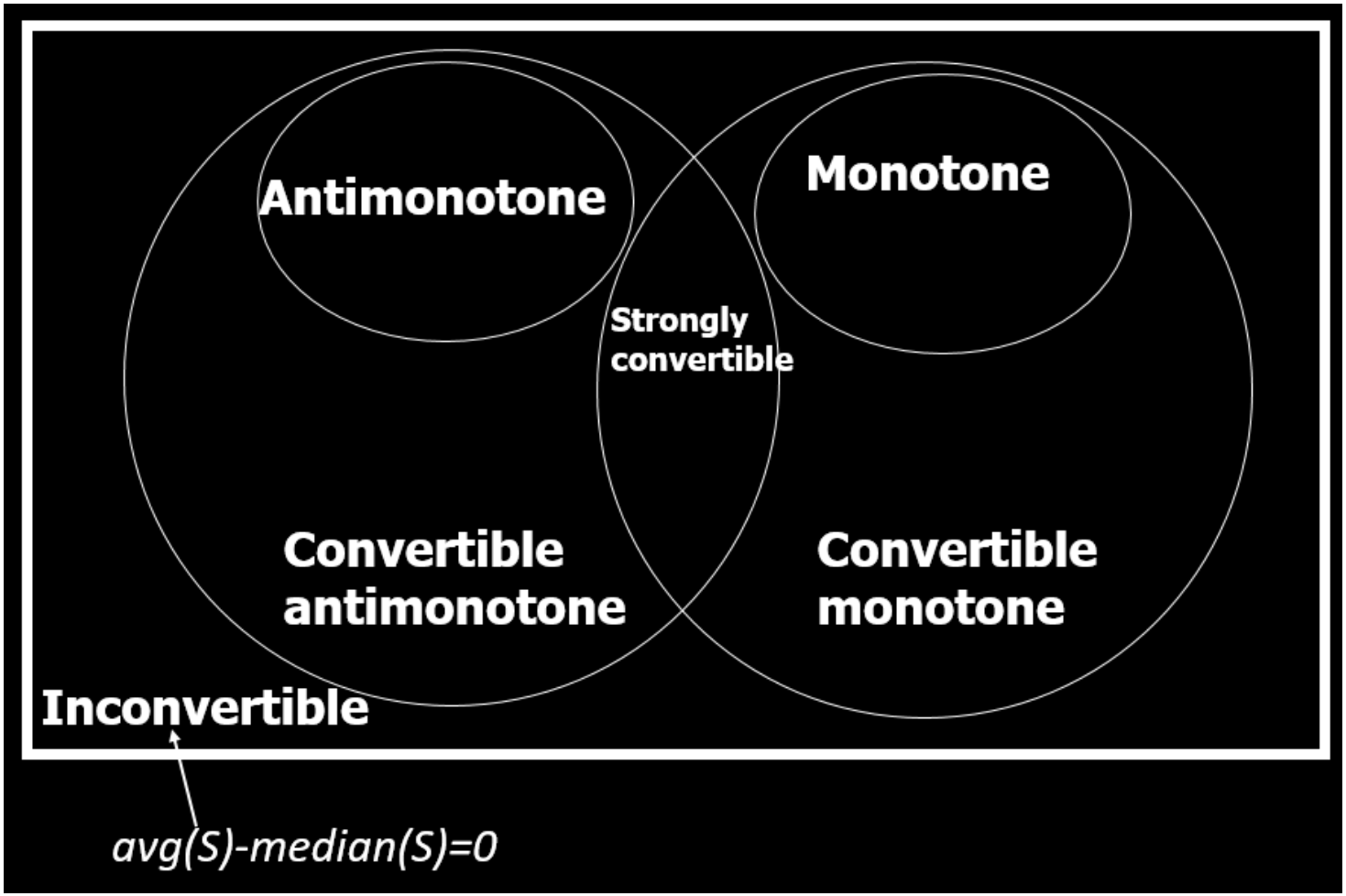
Convertible Monotone and Antimonotone Constraint

- $avg(S) \leq v$ and $avg(S) \geq v$ are neither monotone nor antimonotone.
- A constraint C is convertible monotone when there exists an item order R such that for any itemset A respecting R we have that if $C(A) = true$ then $C(B) = true$ where B is any itemset respecting R and such that A is a suffix of B. For instance:
  - $avg(S) \geq v$ with respect to decreasing price order.
- A constraint C is convertible antimonotone when there exists an item order R such that for any itemset A respecting R we have that if $C(A) = true$ then $C(B) = true$ where B is any itemset respecting R and such that B is a suffix of A. For instance:
  - $avg(S) \geq v$ with respect to increasing price order.
- Alternatively, if $C(B) = false$ for some itemset B respecting R such that B is a suffix of A then $C(A) = false$.
- A constraint that is both convertible monotone and antimonotone is called strongly convertible.

Convertible Monotone and Antimonotone Constraints

Constraint	Convertible antimonotone	Convertible monotone	Strongly convertible
$avg(S) \leq , \geq v$	Yes	Yes	Yes
$median(S) \leq , \geq v$	Yes	Yes	Yes
$sum(S) \leq v$ (items could be of any value, $v \geq 0$)	Yes	No	No
$sum(S) \leq v$ (items could be of any value, $v \leq 0$)	No	Yes	No
$sum(S) \geq v$ (items could be of any value, $v \geq 0$)	No	Yes	No
$sum(S) \geq v$ (items could be of any value, $v \leq 0$)	Yes	No	No
.....			

Summary


Antimonotone
Monotone
Strongly convertible
Convertible antimonotone
Convertible monotone
Inconvertible
avg(S)-median(S)=0