

# Lecture 14

## Association Rules

**Giuseppe Manco**

*Readings:*

Chapter 6, Han and Kamber

Chapter 14, Hastie , Tibshirani and Friedman

# Association Rule Mining

- Dato un insieme di transazioni, trovare le regole che predicono l'occorrenza di un item sulla base delle occorrenze di altri items nella transazione
- Conosciuta anche come **market basket analysis**

## Transazioni Market-Basket

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

## Esempi di associazioni

{pannolini} → {birra},  
{latte, Bread} → {Eggs, Coke},  
{birra, Bread} → {latte},

NB: l'implicazione è co-occorrenza, non causalità!

# Il contesto

Abitudini del cliente tramite l'analisi delle correlazioni tra le varie cose che il cliente acquista



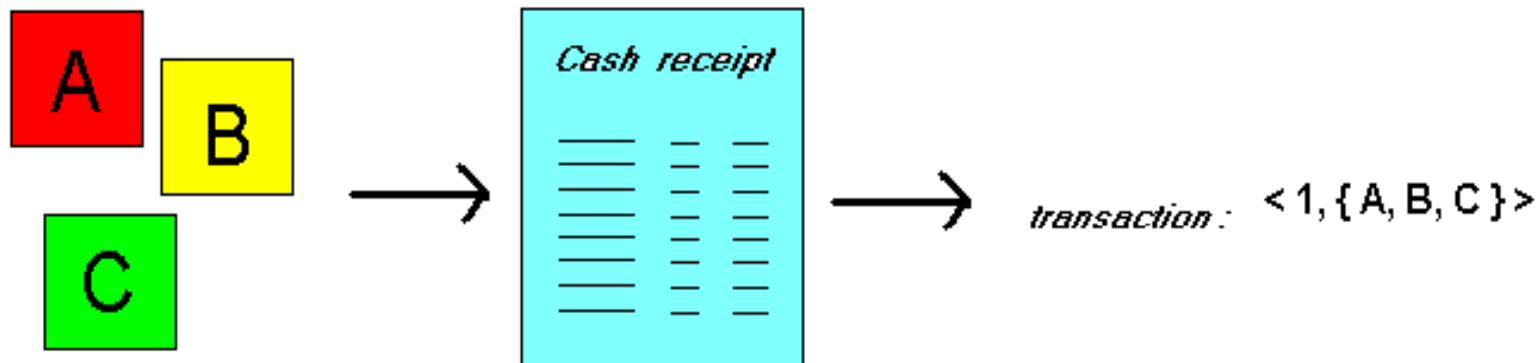
## Il contesto [2]

- Dato:

- Un database di transazioni, dove ogni transazione è un insieme di items

- Trovare:

- I gruppi di items che sono stati acquistati insieme frequentemente



# Road Map

- **AR Unidimensionali/multidimensionali**
  - Su insieme o su attributi
  - Intra-Attributo, Inter-Attributo
- **AR Qualitative/quantitative**
  - Dati categorici, dati numerici
- **AR semplici/basate su vincoli**
  - Esempio: acquisti piccoli ( $\text{sum} < 100$ ) causano grandi acquisti ( $\text{sum} > 1,000$ )
- **Single level/multiple-level AR**
  - Esempio Quali marche di birra sono associate con quali marche di pannolini?
- **Associazioni/correlazioni**
  - Causalità

# Definizioni base

- **Itemset**
  - Una collezione di items
    - Example: {latte, Bread, pannolini}
  - k-itemset
    - Un itemset contenente k items
- **Support ( $\sigma$ )**
  - Frequenza di occorrenze di un itemset
  - E.g.  $\sigma(\{\text{latte, Bread, pannolini}\}) = 2$
- **Itemset frequente**
  - Un itemset il cui supporto è maggiore di un valore soglia

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

# Definizioni base [2]

- **Transazione**

- **Formato relazionale**

Transaction ID	Items
1	formaggio
1	frutta
2	formaggio
2	frutta
2	insalata
3	formaggio
4	frutta
4	cereali

- **Formato colonnare**

Transaction ID	formaggio	frutta	insalata	cereali
1	1	1	0	0
2	1	1	1	0
3	1	0	0	0
4	0	1	0	1

- **Formato compatto**

Transaction ID	Items
1	formaggio,frutta
2	formaggio,frutta,insalata
3	formaggio
4	frutta,cereali

# Itemset frequenti

Transaction ID	Items
1	formaggio, frutta
2	formaggio, frutta, insalata
3	formaggio
4	frutta, cereali

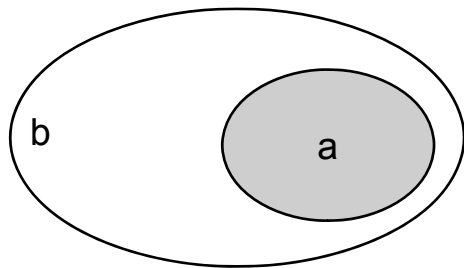
- $\text{Support}(\{\text{formaggio}\}) = 3$  (75%)
- $\text{Support}(\{\text{frutta}\}) = 3$  (75%)
- $\text{Support}(\{\text{formaggio}, \text{frutta}\}) = 2$  (50%)

• Se  $\sigma = 60\%$

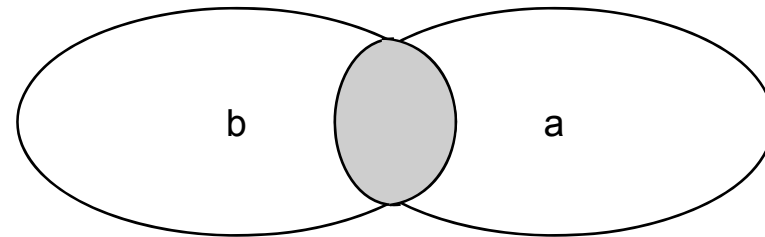
•  $\{\text{formaggio}\}$  e  $\{\text{frutta}\}$  sono frequenti, mentre  $\{\text{formaggio}, \text{frutta}\}$  non lo è.

# Itemset frequenti, regole logiche

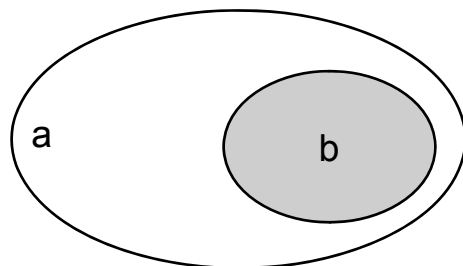
- **La co-occorrenza non implica causalità**



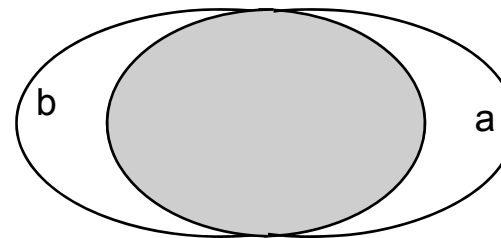
$a \Rightarrow b$



Nessuna relazione



$b \Rightarrow a$



Sia  $b \Rightarrow a$  che  $a \Rightarrow b$

# Definizioni base [3]

- **Regola associativa**
  - Un'implicazione della forma  $X \rightarrow Y$ , dove  $X$  e  $Y$  sono itemsets
  - Esempio:  
 $\{\text{latte, pannolini}\} \rightarrow \{\text{birra}\}$
- **Metriche per la valutazione di una regola**
  - **Support (s)**
    - Frazione delle transazioni che contengono  $X$  e  $Y$
  - **Confidence (c)**
    - Misura quante volte  $Y$  appare nelle transazioni che contengono  $X$

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

**Esempio:**

$\{\text{Milk, Diaper}\} \Rightarrow \text{Beer}$

$$s = \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67$$

# Regole associative e interpretazione probabilistica

$$\text{support}(A \Rightarrow B) = p(A \cup B)$$

$$\text{confidence}(A \Rightarrow B) = p(B|A) = p(A \& B)/p(A).$$

# Il problema

- Dato un insieme  $T$  di transazioni, trovare tutte le regole per cui
  - supporto  $\geq \text{minsup}$
  - confidenza  $\geq \text{minconf}$
- Confidenza alta = forte regolarità
- Supporto alto = il pattern occorre spesso
  - La co-occorrenza non è casuale

# Applicazioni 1 (vendite al dettaglio)

- **market baskets**
  - Le catene di supermercati mantengono le transazioni relative agli acquisti dei clienti
  - conseguenze
    - Conoscere le abitudini d'acquisto dei clienti
    - Posizionamento adeguato dei prodotti
    - Cross-selling –gli hamburger al saldo, il prezzo del ketchup aumentato
    - ...

# Applicazioni 2 (Information Retrieval)

- **Scenario 1**
  - baskets = documenti
  - items = parole
  - Gruppi di parole frequenti = concetti correlati.
- **Scenario 2**
  - items = frasi
  - baskets = documenti contenti frasi
  - Gruppi di frasi frequenti = possibili plagii

# Applicazione 3 (Web Search)

- **Scenario 1**
  - baskets = pagine web
  - items = link in uscita
  - Pagine con riferimenti simili → stessi topics
- **Scenario 2**
  - baskets = pagine web
  - items = link in entrata
  - Pagine con gli stessi in-links → mirrors

# Regole associative

<i>TID</i>	<i>Items</i>
1	pane, latte
2	pane, pannolini, birra, uova
3	latte, pannolini, birra, fanta
4	pane, latte, pannolini, birra
5	pane, latte, pannolini, fanta

## Esempi di regole:

{latte, pannolini} → {birra} (s=0.4, c=0.67)

{latte, birra} → {pannolini} (s=0.4, c=1.0)

{pannolini, birra} → {latte} (s=0.4, c=0.67)

{birra} → {latte, pannolini} (s=0.4, c=0.67)

{pannolini} → {latte, birra} (s=0.4, c=0.5)

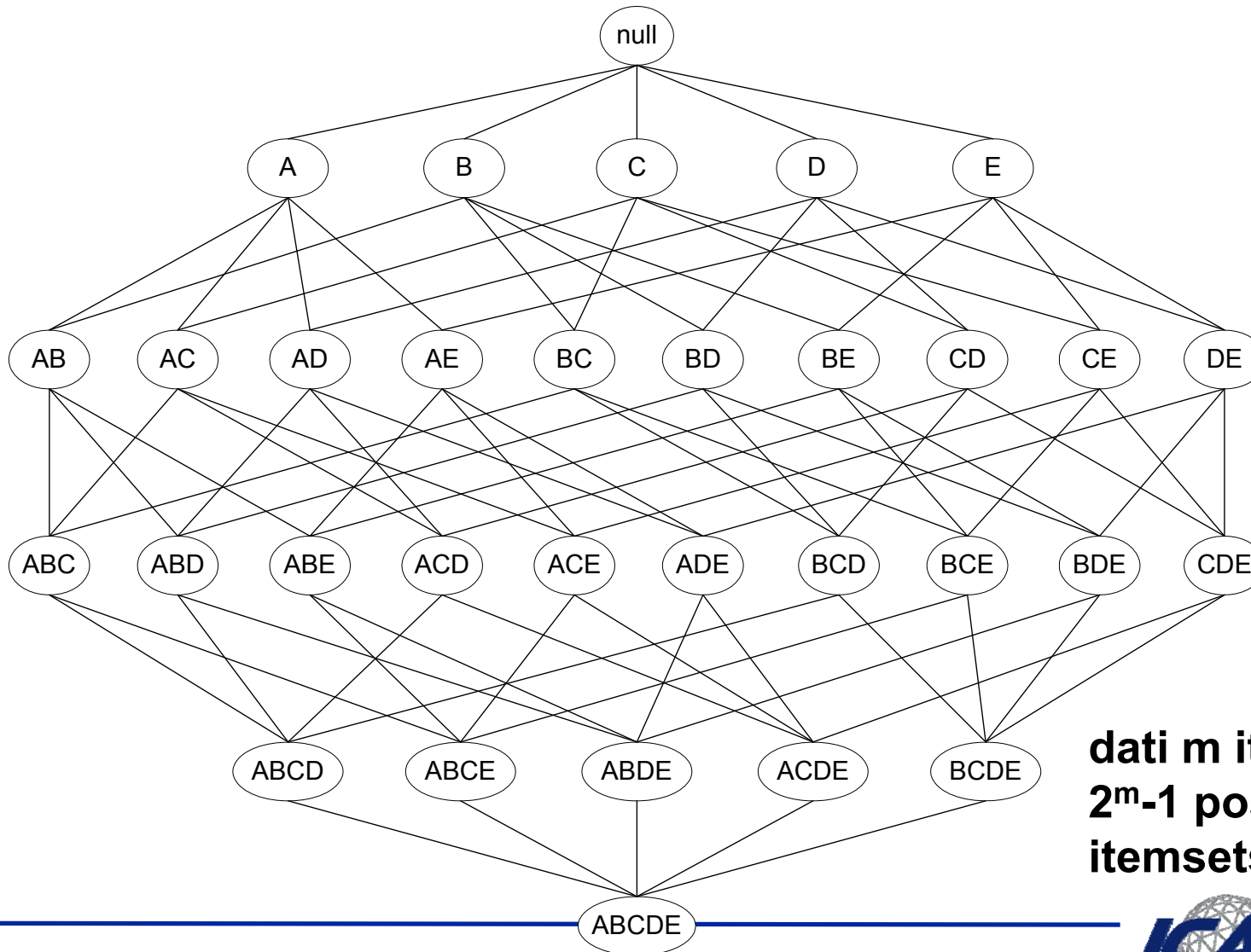
{latte} → {pannolini, birra} (s=0.4, c=0.5)

- **Le regole sono ottenute dal dataset {latte, pannolini, birra}**
  - Supporto simile, confidenza differente
- **Decomposizione del problema:**
  - Trovare tutti gli itemset frequenti
  - Utilizzare gli itemset frequenti per trovare le regole

# Mining di regole associative

- **Obiettivo** – trovare tutte le regole per cui
  - Supporto  $\geq s$
  - confidenza  $\geq c$
- **Riduzione del problema**
  - trovare tutti gli itemsets frequenti  $X$
  - Dato  $X=\{A_1, \dots, A_k\}$ , generare le regole  $X-A_j \rightarrow A_j$
  - Confidenza =  $\text{sup}(X)/\text{sup}(X-A_j)$
  - Supporto =  $\text{sup}(X)$
  - Eliminiamo le regole il cui supporto è basso
- **Problema principale**
  - Trovare gli itemsets frequenti

# Il reticolo degli itemsets



**dati m items, ci sono  
 $2^m - 1$  possibili  
itemsets candidati**

# La scala del problema

- **Catena di supermercati**
  - vende  $m=100,000$  items
  - traccia  $n=1,000,000,000$  transazioni al giorno
- **Web**
  - Miliardi di pagine
  - Approssimativamente una parola diversa per ogni pagina
- **Un numero esponenziale di itemsets**
  - $m$  items  $\rightarrow 2^m - 1$  possibili itemsets
  - Non possiamo considerare tutti gli itemsets dato  $m$
  - Anche i 2-itemsets possono essere troppi
    - $m=100,000 \rightarrow 5.000$  miliardi di coppie

# Utilizzando SQL

- Assunzione di base: formato relazionale
  - Purchase(TID, ItemID)

- 3-itemsets:

```
SELECT Fact1.ItemID, Fact2.ItemID,  
       Fact3.ItemID, COUNT(*)  
FROM Purchase Fact1  
JOIN Purchase Fact2 ON Fact1.TID = Fact2.TID  
                   AND Fact1.ItemID < Fact2.ItemID  
JOIN Purchase Fact3 ON Fact1.TID = Fact3.TID  
                   AND Fact1.ItemID < Fact2.ItemID  
                   AND Fact2.ItemID < Fact3.ItemID  
GROUP BY Fact1.ItemID, Fact2.ItemID, Fact3.ItemID  
HAVING COUNT(*) > 1000
```

- Trovare i k-itemsets richiede k operazioni di join!

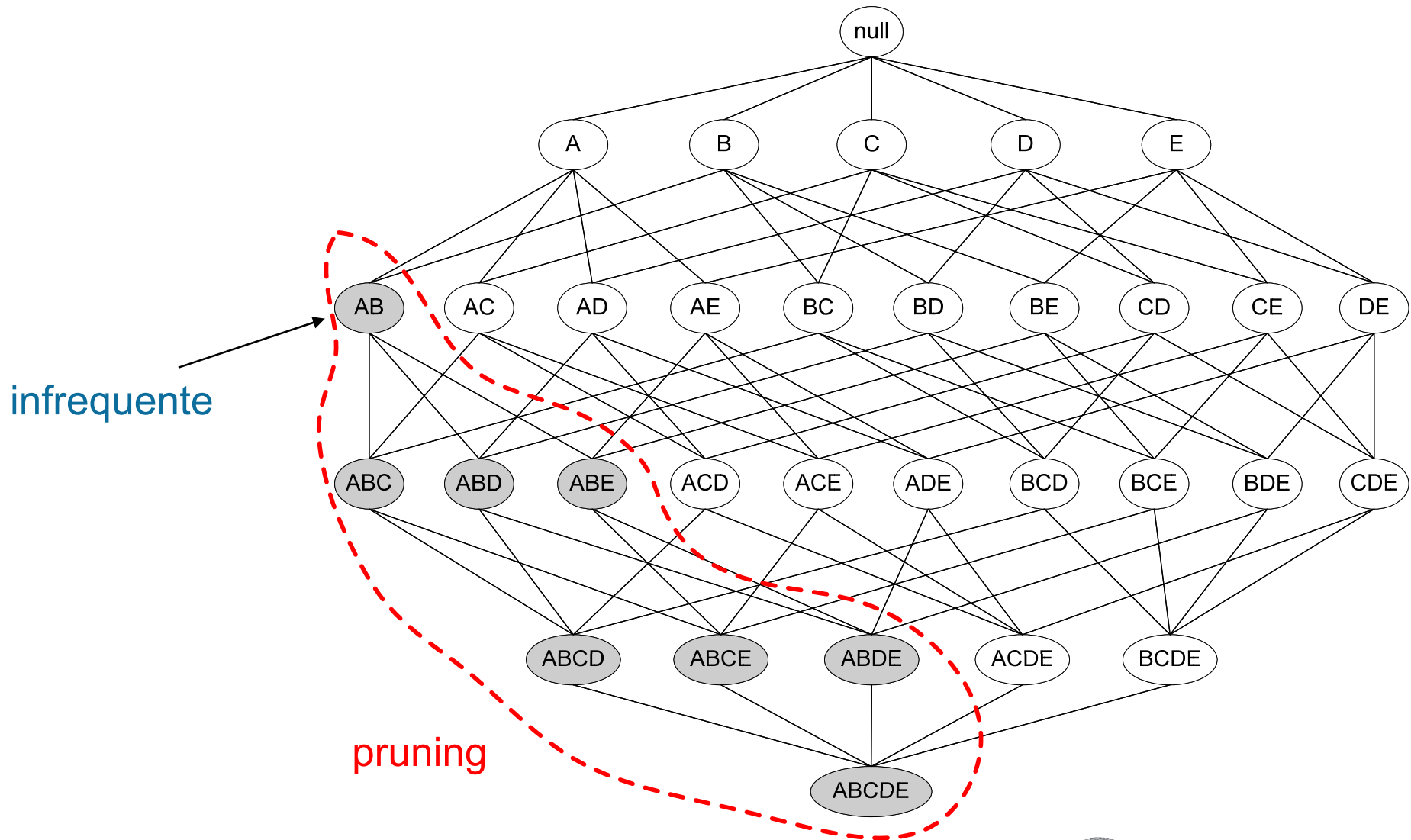
# Monotonicità

- **Idea chiave:**
  - Se un itemset è **frequente**,
  - Tutti i suoi **sottoinsiemi** devono essere **frequenti**
- Il principio di monotonicità vale per il supporto:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

- **Strategia di pruning:**
  - Se un itemset è **infrequente**, tutti i suoi **sovrainsiemi** devono essere **infrequenti**

# Monotonicità



# L'algoritmo Apriori

- Join Step
  - $C_k$  è generato fondendo  $L_{k-1}$  con sé stesso
- Prune Step
  - Tutti i (k-1)-itemsets non frequenti non possono essere sottoinsiemi di un k-itemset frequente
- Pseudo-codice:
  - $C_k$ : itemsets candidati di dimensione k
  - $L_k$ : itemsets frequenti di dimensione k

```
1.  $L_1 = \{\text{items frequenti}\};$   
2. for ( $k = 1; L_k \neq \emptyset; k++$ ) do begin  
3.      $C_{k+1} =$  candidati generati da  $L_k$ ;  
4.     for each transazione  $t$  in  $D$   
5.         Incrementa il supporto dei candidati in  $C_{k+1}$  contenuti in  $t$   
6.      $L_{k+1} =$  tutti i candidati in  $C_{k+1}$  con  $\text{min\_support}$   
7. return  $\cup_k L_k$ ;
```

# Esempio

Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D

$C_1$

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

$L_1$

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

$C_2$

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

Scan D

$C_2$

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

$L_2$

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

$C_3$

itemset
{2 3 5}

Scan D

$L_3$

itemset	sup
{2 3 5}	2

Association rules mining

# Generazione di candidati [1]

- **Fase di join**
  - **Passo 3**

$$\left. \begin{array}{l} p = \{a, b, c\} \in L_k \\ q = \{a, b, d\} \in L_k \end{array} \right\} r = \{a, b, c, d\} \in C_{k+1}$$

insert into C<sub>k+1</sub>

select p.item<sub>1</sub>, p.item<sub>2</sub>, ..., p.item<sub>k-1</sub>, q.item<sub>k-1</sub>

from p, q

where p.item<sub>1</sub>=q.item<sub>1</sub> and p.item<sub>2</sub>= q.item<sub>2</sub> and ... p.item<sub>k-2</sub>= q.item<sub>k-2</sub>

and p.item<sub>k-1</sub> < q.item<sub>k-1</sub>

# Generazione candidati [2]

- **Fase di pruning**
  - **Passo 3**

$$L_3 = \{\{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{a, c, e\}, \{b, c, d\}\}$$

- **Inizialmente,**

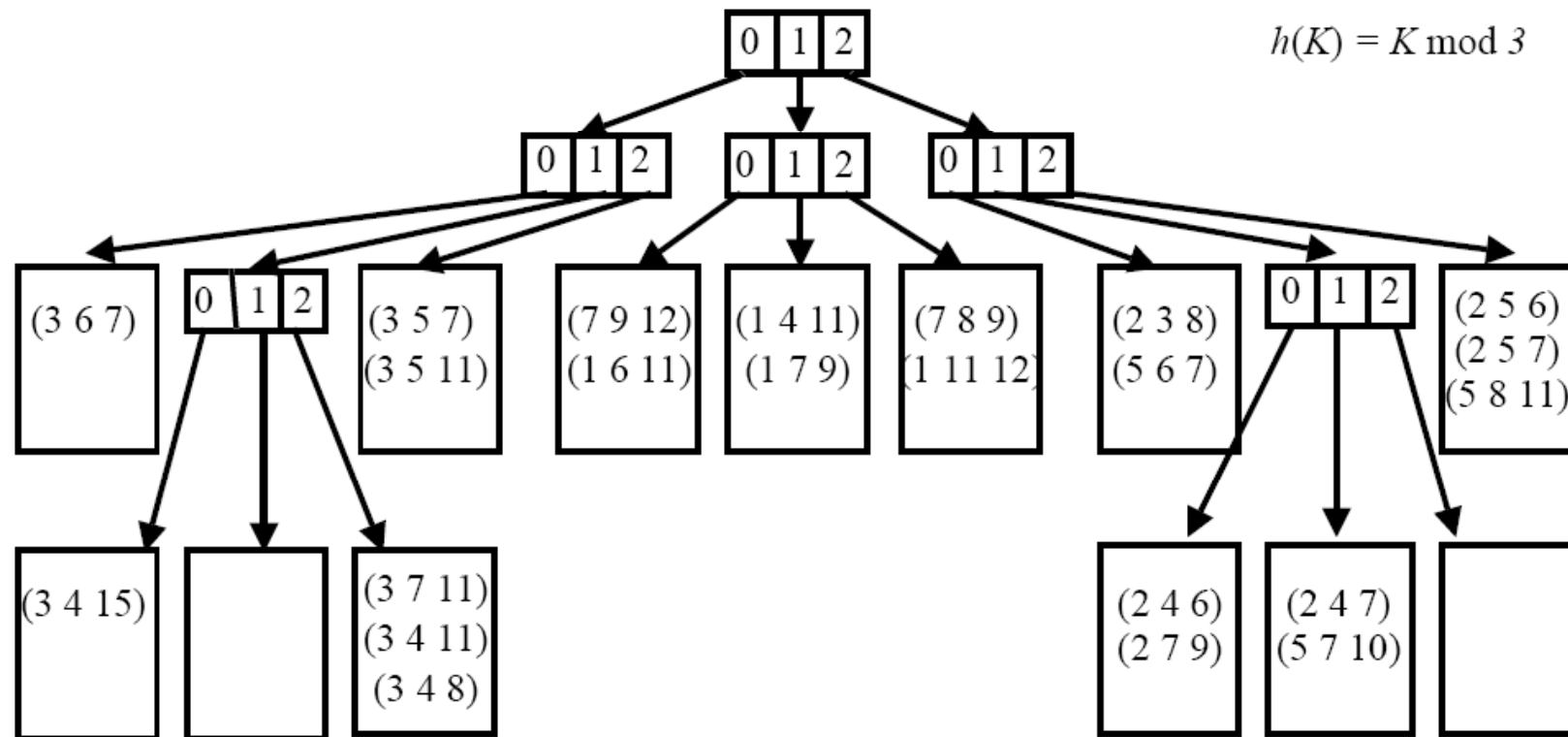
$$C_4 = \{\{a, b, c, d\}, \{a, c, d, e\}\}$$

- **Eliminiamo  $\{a, c, d, e\}$** 
  - $\{c, d, e\} \notin L_3$

# Generazione degli itemsets frequenti

- **Conteggio dei candidati**
  - **Passo 6**
- **Perché è problematico?**
  - **Il numero totale dei candidati può essere alto**
  - **Una transazione può contenere molti candidati**
- **Soluzione**
  - **Hash-tree**
  - **Nodi foglia**
    - **Itemsets candidati**
  - **Nodi interni**
    - **Tabelle hash**
      - **Ogni bucket punta ad un sottoinsieme di candidati**
  - **Speed-up della relazione di sottoinsieme**

# Hash-tree

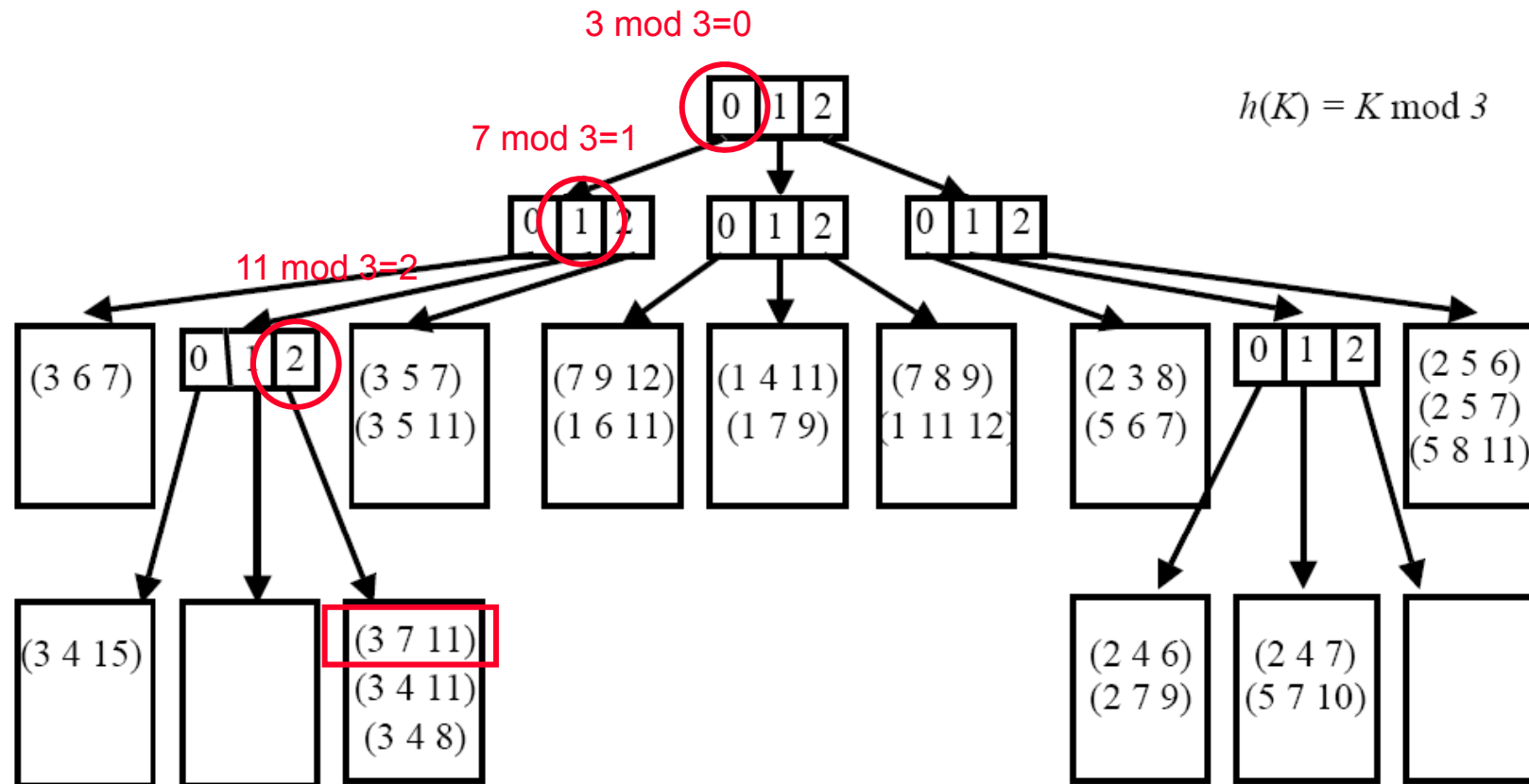


# Utilizzo dell'Hash-tree

- **Inserzione**
  - Identifica la foglia opportuna e inserisci l'itemset
    - La foglia è ottenuta con la ricerca
    - Se c'è overflow, splitta la foglia in due foglie e aggiungi un nuovo nodo interno
- **Ricerca**
  - Parti dalla radice
  - Ad ogni livello  $i$ :
    - Applica la funzione hash all' $i$ -esimo elemento dell'itemset
    - Scendi al livello puntato dal risultato della funzione
- **Counting**
  - Passo 5
  - $t = \{t_1, t_2, \dots, t_n\}$
  - Ad ogni livello  $d$ 
    - Determina il nodo puntato da  $t_i$
    - Continua la ricerca con  $\{t_{i+1}, \dots, t_n\}$
  - Se nodo foglia
    - Aggiorna il supporto degli itemsets contenuti in  $t$

# Ricerca

- {3,7,11}**



# Counting

- $T = \{1, 3, 7, 9, 12\}$

$$h(K) = K \bmod 3$$

