Regole associative

Prof. Matteo Golfarelli
Alma Mater Studiorum - Università di Bologna

Mining di regole associative

- Dato un insieme di transazioni, trovare le regole che segnalano la presenza di un elemento sulla base della presenza di altri elementi nella transazione

Transazioni del carrello della spesa "Market-Basket"

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Alcuni esempi…

{Diaper} → {Beer},
{Milk, Bread} → {Eggs,Coke},
{Beer, Bread} → {Milk},

L'implicazione implica co-occorrenza non causalità!

ATTENZIONE: gli item sono modellati da variabili binarie asimmetriche. Un item è presente oppure è assente nella transazione; la sua presenza è considerata un evento più importante della sua assenza

Frequent Itemset - Insiemi Frequenti

- Itemset
  - ✓ Una collezione di uno o più elementi
    - Esempi: {Milk, Bread, Diaper}
  - ✓ k-itemset
    - Un itemset che contiene k-elementi
- **Conteggio del supporto – σ( )**
  - ✓ Numero di istanze dell'itemset nell'insieme di transazioni
  - ✓ Es. σ({Milk, Bread,Diaper}) = 2
- **Supporto – s( )**
  - ✓ Frazione delle transazioni che contiene l'itemset
  - ✓ Es. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - ✓ Un itemset il cui supporto è maggiore o uguale a una soglia *minsup*

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Regole associative

- Regole associative
  - ✓ Una implicazione con forma X → Y, dove X e Y sono itemset
  - ✓ Es. {Milk, Diaper} → {Beer}
- Metriche per la valutazione delle regole associative
  - ✓ Supporto (s)
    - Frazione delle transazioni che includono X e Y
  - ✓ Confidenza (c)
    - Misura quante volte gli elementi in Y appaiono in transazioni che contengono X

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Esempio:

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

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

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

Supporto e Confidenza: interpretazione statistica

- Il supporto indica la probabilità della presenza di X e Y nella transazione

$$P(X,Y) = \frac{\sigma(X,Y)}{N}$$

- La confidenza per la regola X→Y modella la probabilità della presenza di Y nella transazione condizionata alla presenza di X

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)} = \frac{\sigma(X,Y)}{N}\frac{N}{\sigma(X)} = \frac{\sigma(X,Y)}{\sigma(X)}$$

Esercizio

- Date le seguenti transazioni calcolare:
  - ✓ Il supporto per gli itemset {e}, {b,d}, {b,d,e}
  - ✓ Calcolare la confidenza per le regole {b,d} → {e} {e} → {b,d}

Transazione	Item
1	{a,d,e}
2	{a,b,c,e}
3	{a,b,d,e}
4	{a,c,d,e}
5	{b,c,e}
6	{b,d,e}
7	{c,d}
8	{a,b,c}
9	{a,d,e}
10	{a,b,e}

Formulazione del problema

- Dato un insieme di transazioni T, si vuole trovare tutte le transazioni tali che:
  - ✓ Supporto ≥ *minsup*
  - ✓ Confidenza ≥ *minconf*

- Approccio naive:
  - ✓ Elenca tutte le possibili regole associative
  - ✓ Per ogni regola calcola il supporto e la confidenza
  - ✓ Elimina le regole che non superano le soglie per *minsup* e *minconf*

Computazionalmente proibitivo!!

Scoperta di regole associative

TID	Items
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 regole

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Osservazione:

• Tutte le regole sono partizioni binarie dello stesso itemset:
{Milk, Diaper, Beer}

• Le regole basate sullo stesso itemset hanno sempre il medesimo *supporto* ma possono avere *confidenze* diverse

Quindi è possibile disaccoppiare il calcolo della confidenza e del supporto

Scoperta di regole associative

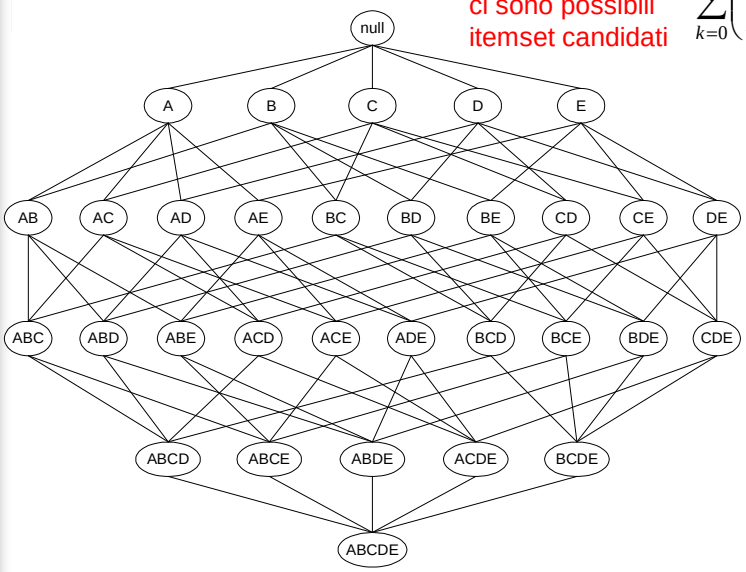
- Un approccio in due fasi:
  1. Generazione degli Itemset frequenti
     - Generare tutti gli itemset con supporto ≥ minsup
  2. Generazione delle regole
     - Per ogni itemset frequente, generare le regole con confidenza elevata. Ogni regola è un partizionamento binario degli elmenti nell'itemset

La generazione degli itemset frequenti è comunque un problema computazionalemente complesso

Generazione degli itemset frequenti

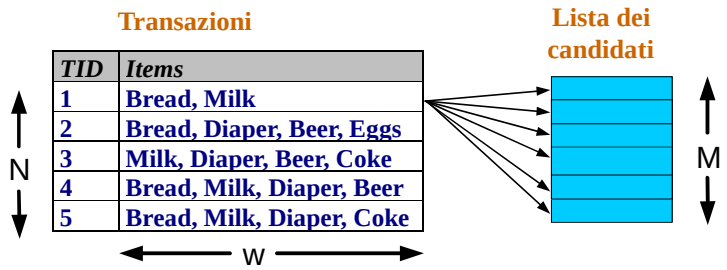
Dati *d* elementi, ci sono possibili itemset candidati

$$\sum_{k=0}^{d} \binom{d}{k} = 2^d$$



Generazione degli Itemset frequenti

- Approccio naive:
  - ✓ Ogni itemset nel reticolo è candidato a essere frequente
  - ✓ Calcola il supporto per ogni candidato scorrendo il database

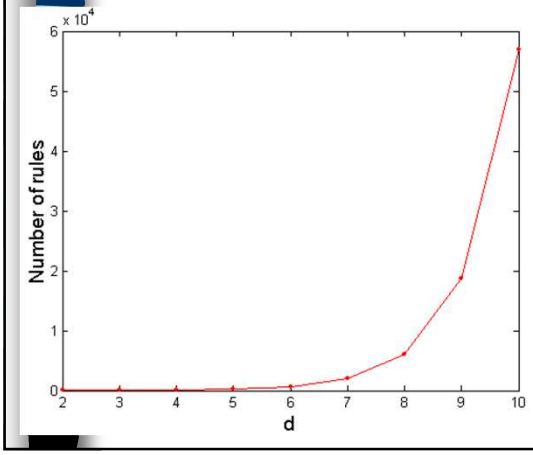


TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

  - ✓ Confronta ogni transazione con ogni candidato
  - ✓ Complessità ~ O(NMw) => Costoso dato che M = 2^d !!!

Complessità computazionale

- Dati d item distinti:
  - ✓ Il numero totale di itemset 2^d
  - ✓ Il numero totale di regole associative *R* è:



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$
$$= 3^d - 2^{d+1} + 1$$

Se d=6, R = 602 rules

Strategie per la generazione di itemset frequenti

- Ridurre il numero dei candidati (M) utilizzando tecniche di pruning
- Ridurre il numero delle transazioni (N) quando il numero degli itemset è troppo elevato
- Ridurre il numero delle comparazioni (NM) utilizzando strutture dati efficienti per memorizzare i candidati o le transazioni
  - ✓ Può non essere necessario verificare ogni candidato con ogni transazione

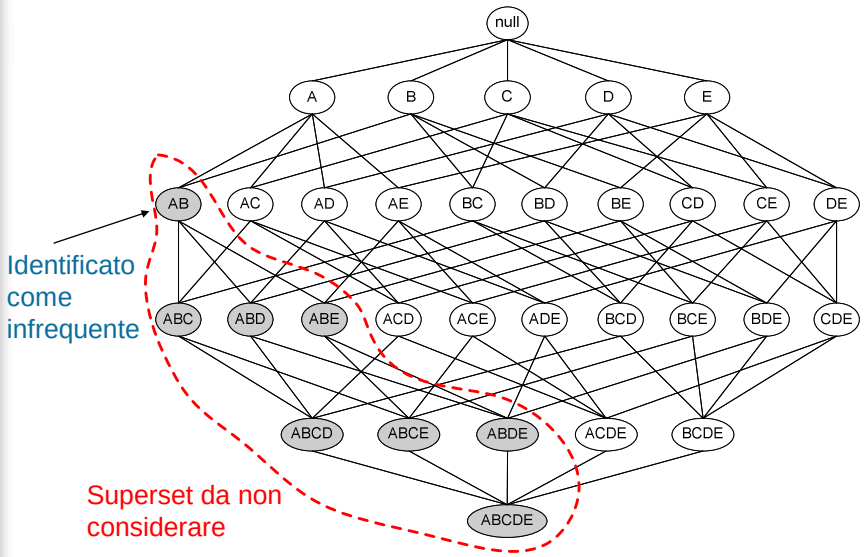
Riduzione del numero dei candidati

- Principio “Apriori”:
  - ✓ Se un itemset è frequente, allora anche tutti i suoi sotto-insiemi devono esserlo.
- Il principio Apriori è dovuto alla seguente proprietà del supporto:

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

  - ✓ Il supporto di un itemset non eccede il supporto dei suoi sottoinsiemi
  - ✓ Questa è nota come proprietà anti-monotona del supporto

Visualizzazione del principio Apriori



Visualizzazione del principio Apriori

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Item (1-itemsets)

Minimum Support = 3

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Coppie (2-itemsets)

(Non è necessario generare i candidati che coinvolgono Coca o uova)

Terne (3-itemsets)

Itemset	Count
{Bread,Diaper, Milk }	3

Considerando tutti gli itemset

$$\binom{6}{1}+\binom{6}{2}+\binom{6}{3}=6+15+20=41$$

Applicando il pruning basato sul supporto,

$$\binom{6}{1}+\binom{4}{2}=6+6+1=13$$

Algoritmo Apriori

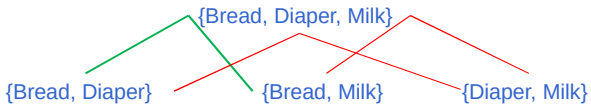
```
k=1
F_k = { i | i ∈ I ∧ σ({i}) ≥ N × minsupp}
// trova gli 1-itemset frequenti
repeat
  k=k+1
  C_k=apriori-gen(F_k-1) // genera gli itemset candidati
   for each transazione t ∈ F_k
     C_t = subset(C_k,t) // determina gli itemset che compaiono in t
     for each itemset candidato c ∈ C_t
      σ(c) = σ(c)+1 // incrementa il supporto
   end for
  end for
  F_k = {c | c ∈ C_k ∧ σ(c) ≥ N × minsupp}
  // identifica i k-itemset frequenti
until F_k = ∅
Risultato = ∪F_k
```

Generazione degli itemset candidati: apriori-gen(..)

- La procedura apriori-gen svolge le seguenti operazioni:
  - ✓ Genera i k-itemset candidati
    - Il numero di k-itemset generabili a partire da d item è $\binom{d}{k}$
  - ✓ Esegue il pruning dei candidati sulla base del principio del supporto
    - Tutti i (k-1)-itemset del k-itemset devono essere frequenti perchè il k-itemset sia frequente
- Ci sono più modi per generare gli itemset candidati. Un buon approccio deve garantire:
  - ✓ Evitare la generazione di itemset “inutili”, ossia itemset che saranno eliminati tramite il pruning perchè non frequenti
  - ✓ Deve essere completo, ossia deve generare tutti gli itemset che frequenti
  - ✓ Deve essere efficiente, per esempio non deve generare un itemset più volte

Generazione degli itemset candidati: apriori-gen(..)

- **Fk-1 x Fk-1:** l'approccio utilizzato nell'algoritmo apriori fonde due k-1 itemset A e B solo se i relativi (k-2)-itemset sono uguali:
  - ✓ $A=\{a_1,\ldots,a_{k-1}\}$ $B=\{b_1,\ldots,b_{k-1}\}$ $a_i=b_i \ \forall\ i=1..k-2 \wedge a_{k-1}\neq b_{k-1}$
- E' necessario mantenere gli item ordinati lessicograficamente
- Si evita di generare più volte lo stesso itemset



- La generazione è completa (nessun k-itemset potenzialmente frequente è tralasciato)
  - ✓ Se un k-itemset è frequente lo devono essere anche tutti i suoi (k-1)-itemset, quindi anche quelli utilizzati per la generazione
- Sebbene il k-itemset sia generato fondendo due (k-1)-itemset frequenti, è necessario verificare che i restanti k-2 sottoinsiemi siano frequenti

Esercizio

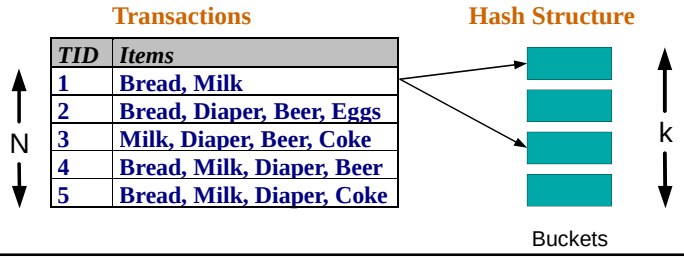
- Dati i seguenti 3-itemset frequenti
  - ✓ Elencare i 4-itemset candidati ottenuti utilizzando Fk-1 x Fk-1
  - ✓ Quali candidati sopravvivono alla fase di pruning Apriori

{1, 2, 3} {1, 2, 4} {1, 2, 5} {1, 3, 4} {1, 3, 5}
{2, 3, 4} {2, 3, 5} {3, 4, 5}

- Elencare infine i 4-itemset prodotti utilizzando il metodo Fk-1 xF1 che unisce al (k-1)-itemset frequente un item frequente
  - ✓ Si suponga che esistano solo gli item da 1 a 5

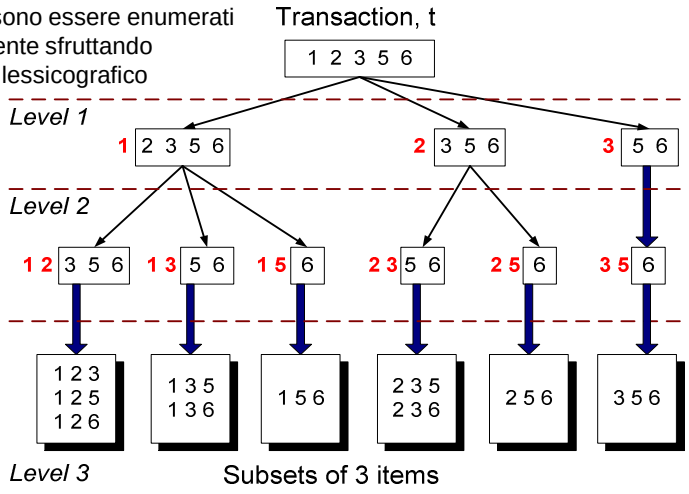
Ridurre il numero delle comparazioni

- Conteggio dei candidati:
  - ✓ Richiede la scansione delle transazioni per determinare il supporto degli itemset candidati
    - Per ogni transazione si generaranno tutti i k-itemset e si procederà a incrementare il supporto dei corrispondenti itemset candidati
  - ✓ Per ridurre il numero delle comparazioni è utile memorizzare i candidati in una struttura hash
    - Invece di eseguire il match di ogni transazione con ogni candidato si eseguirà il match solo con i candidati presenti nel bucket



Enumerazione di tutti gli itemset presenti in una transazione

Data una transazione t, i 3-itemset frequenti possono essere enumerati in modo efficiente sfruttando l'ordinamento lessicografico

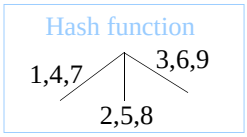


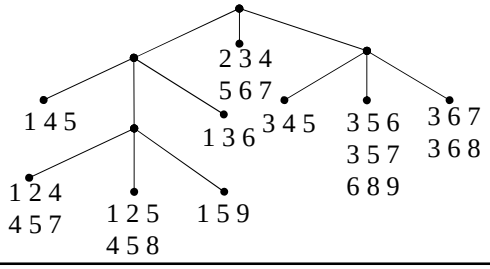
Generazione dell'Hash Tree

- Si supponga di avere15 itemset candidati di lunghezza 3:

{1 4 5}, {1 2 4}, {4 5 7}, {1 2 5}, {4 5 8}, {1 5 9}, {1 3 6}, {2 3 4}, {5 6 7}, {3 4 5}, {3 5 6}, {3 5 7}, {6 8 9}, {3 6 7}, {3 6 8}

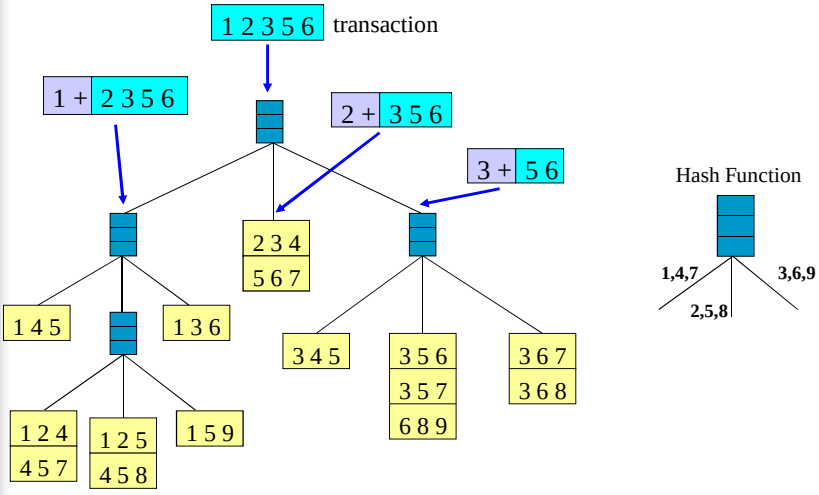
- E' necessario individuare
  - ✓ Una funzione hash
  - ✓ Dimensione massima delle foglie, ossia il numero massimo degli itemset memorizzati in una foglia. Se il numero dei degli item candidati supera la dimensione della foglia esegui lo split del nodo



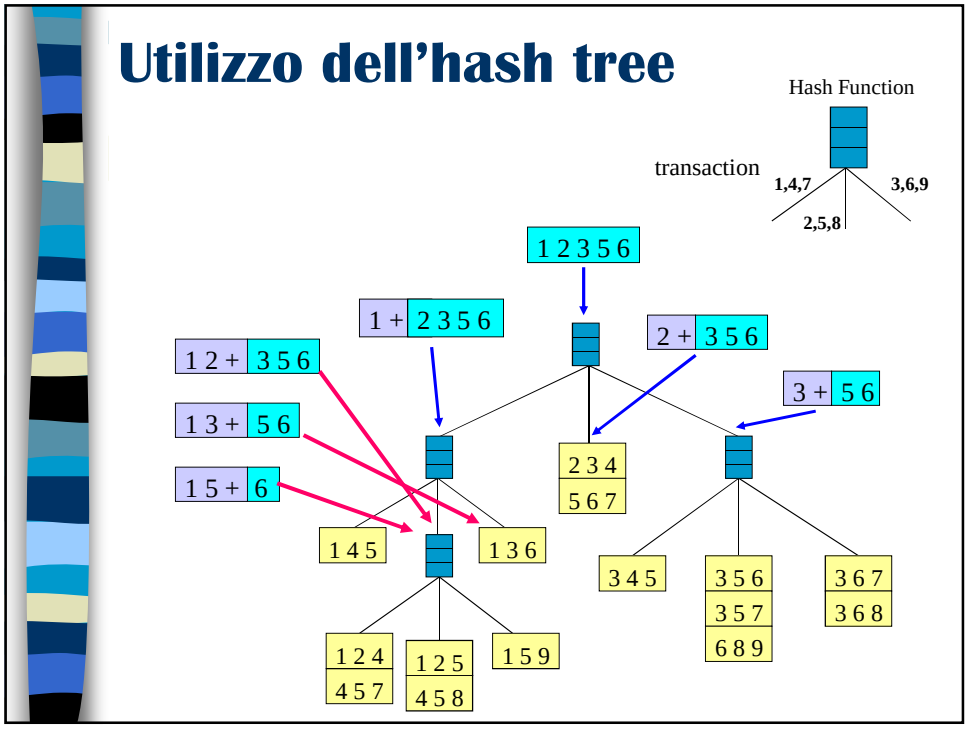


Utilizzo dell'hash tree

- I 3-itemset contenuti nella transazione sono generati e confrontati solo con gli itemset frequenti delle foglie corrispondenti nell'hash tree

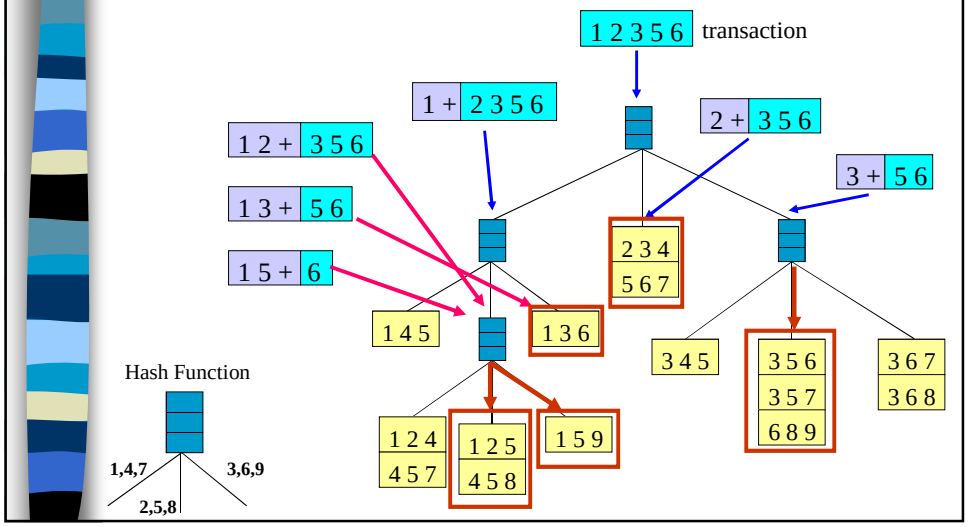


Utilizzo dell'hash tree



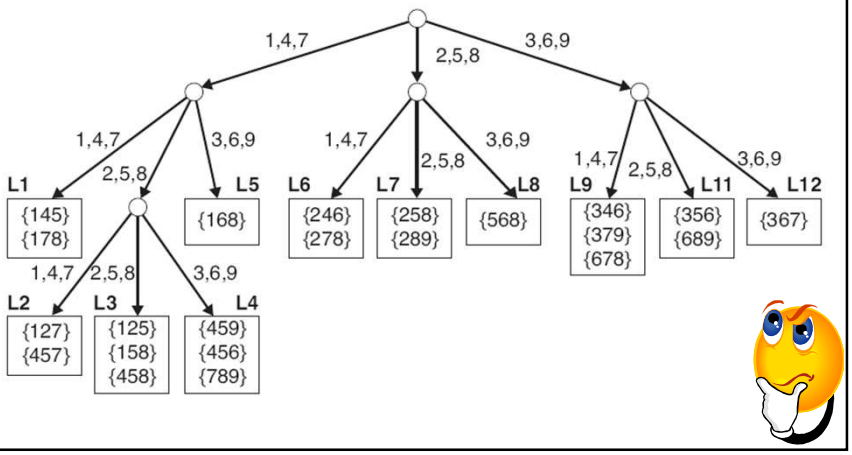
Utilizzo dell'hash tree

- La transazione è confrontata con 11 candidati su 15



Utilizzo dell'hash tree

- Dato il seguente hash-tree per 3-itemset candidati
  - ✓ Valutare quali foglie sono visitate per cercare i candiati della transazione {1,3,4,5,8}
  - ✓ Quali itemset candiati sono contenuti nella transazione?

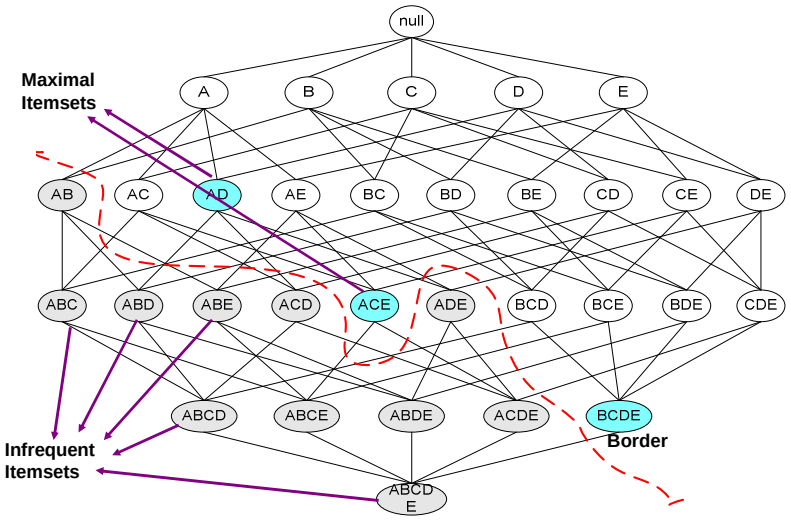


Fattori che influenzano la complessità di Apriori

- Scelta della soglia di supporto minimo
  - ✓ Abbassando il valore di soglia per *minsupp* si otterrà un maggior numero di itemset candidati e potenzialmente aumenterà la lunghezza massima degli itemset frequenti
    - Maggior numero di itemset -> Hash tree più grande
    - Itemset frequenti più lunghi -> Più passate sul dataset
- Numero di transazioni nel database
  - ✓ Visto che Apriori esegue più scansioni del database, il tempo di esecuzione dell'algoritmo è correlato con il numero delle transazioni in esso presenti
- Lunghezza media delle transazioni
  - ✓ Transazioni lunghe tendono a determinare itemset frequenti più lunghi
    - Più spazio richiesto per l'hash tree
    - Più passate sul dataset
- Numero di item
  - ✓ Più spazio è richiesto per il conteggio del supporto

Itemset frequenti massimali

- Un itemset frequente è massimale se nessuno dei sui adiacenti superset è frequente



Itemset chiusi

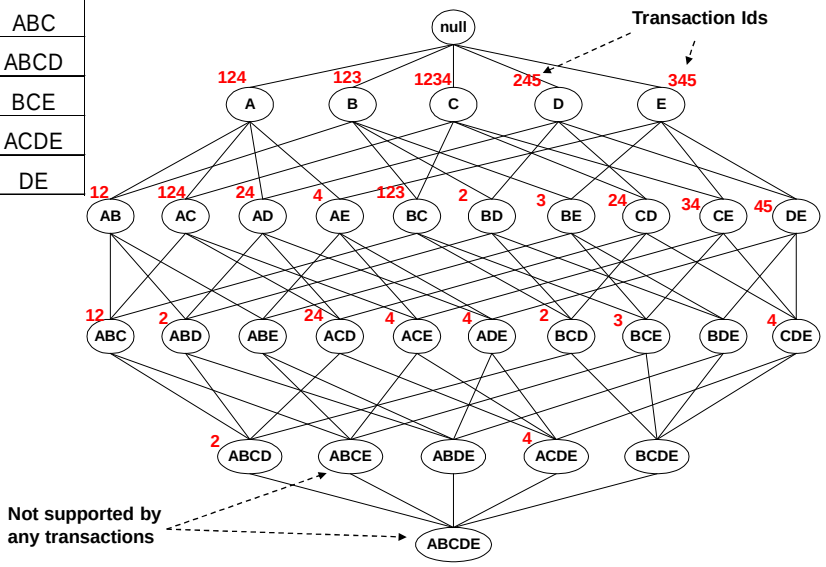
- Un itemset è detto chiuso se nessuno dei sui adiacenti superset ha lo stesso supporto

TID	Items
1	{A,B}
2	{B,C,D}
3	{A,B,C,D}
4	{A,B,D}
5	{A,B,C,D}

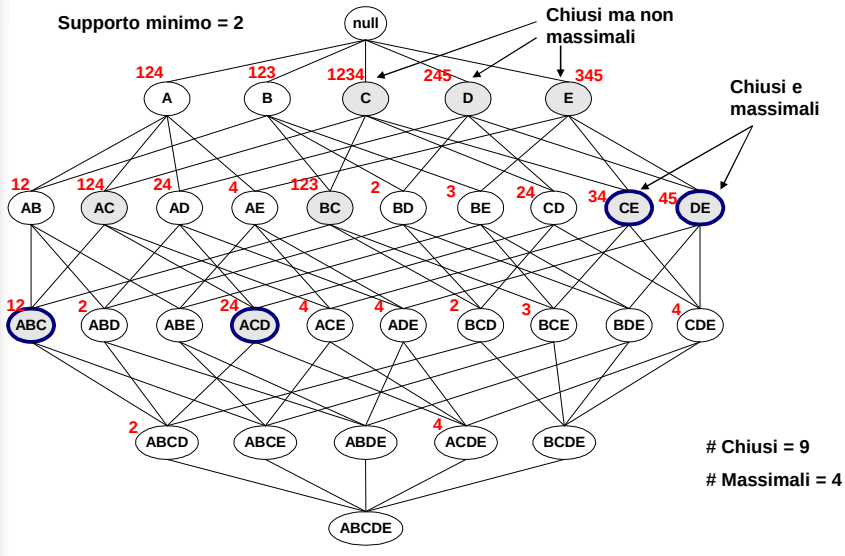
Itemset	Support
{A}	4
{B}	5
{C}	3
{D}	4
{A,B}	4
{A,C}	2
{A,D}	3
{B,C}	3
{B,D}	4
{C,D}	3

Itemset chiusi vs massimali

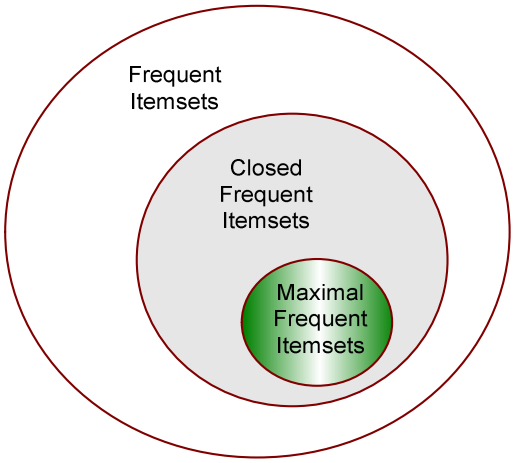
TID	Items
1	ABC
2	ABCD
3	BCE
4	ACDE
5	DE



Itemset chiusi vs massimali



Itemset chiusi vs massimali



Esercizio

- Dato il seguente reticolo e lista di transazioni etichettare i nodi del reticolo:
  - ✓ M maximal frequent
  - ✓ C Closed frequent
  - ✓ F Frequent (non maximal e non closed)
  - ✓ I Infrequent

TID	Item
1	a,b,d,e
2	b,c,d
3	a,b,d,e
4	a,c,d,e
5	b,c,d,e
6	b,d,e
7	c,d
8	a,b,c
9	a,d,e
10	b,d

Generazione delle regole

- Dato un itemset frequente L, trovare tutti i sotto insiemi non vuoti f $\subset$ L such that f $\rightarrow$ L – f che soddisfano il criterio di confidenza minima
  - ✓ Dato l'itemset frequente {A,B,C,D} le regole candidate sono:

ABC $\rightarrow$D,	ABD $\rightarrow$C,	ACD $\rightarrow$B,	BCD $\rightarrow$A,
A $\rightarrow$BCD,	B $\rightarrow$ACD,	C $\rightarrow$ABD,	D $\rightarrow$ABC
AB $\rightarrow$CD,	AC $\rightarrow$ BD,	AD $\rightarrow$ BC,	BC $\rightarrow$AD,
BD $\rightarrow$AC,	CD $\rightarrow$AB,		

- Se |L| = k, allora esistono 2^k – 2 regole associative candidate (escudendo L $\rightarrow \varnothing$ e $\varnothing \rightarrow$ L)

Generazione delle regole

- Come generare le regole in modo efficiente a partire dagli itemset frequenti?
  - ✓ La misura di confidenza non gode della proprietà di anti-monotonicità rispetto alla regola associativa complessiva
    - c(ABC $\rightarrow$D) può essere superiore o inferiore a c(AB $\rightarrow$D)
  - ✓ E' tuttavia possibile sfruttare l'anti-monotonicità della confidenza rispetto alla parte sinistra della regola

Teorema: Dato un itemset Y e due regole r1:X $\rightarrow$ Y-X e r2:X' $\rightarrow$ Y-X' con X' $\subset$ X $\subset$ Y allora c(r1) $\geq$ c(r2)

Dimostrazione: la confidenza delle due regole è definita da:

$$c(r1) = \sigma(Y) / \sigma(X) \qquad c(r2) = \sigma(Y)/ \sigma(X')$$

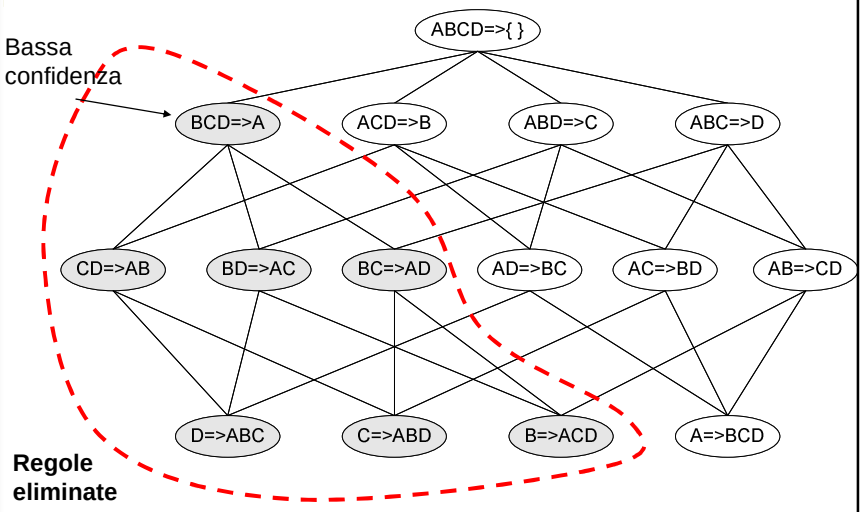
ma visto che X' $\subset$ X sarà $\sigma(X') \geq \sigma(X)$ per la proprietà di anti-monotonocità del supporto

- Esempio: L = {A,B,C,D}

$$c(ABC \rightarrow D) \geq c(AB \rightarrow CD) \geq c(A \rightarrow BCD)$$

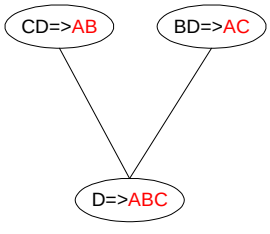
Generazione delle regole

- Reticolo delle regole per l'itemset ABCD



Generazione delle regole

- Per evitare di generare regole con confidenza limitata si utilizza un approccio a livelli: ogni livello è caratterizzato da una cardinalità (crescente) del lato destro delle regole
- Inoltre, per evitare di generare la stessa regola più volte le regole candidate sono generate fondendo due regole che condividono uno stesso prefisso nella parte destra
  - ✓ join(CD → AB,BD → AC) produrranno la regola D → ABC
- La regola D → ABC può essere eliminata se AD → BC non ha una confidenza > *minconf*



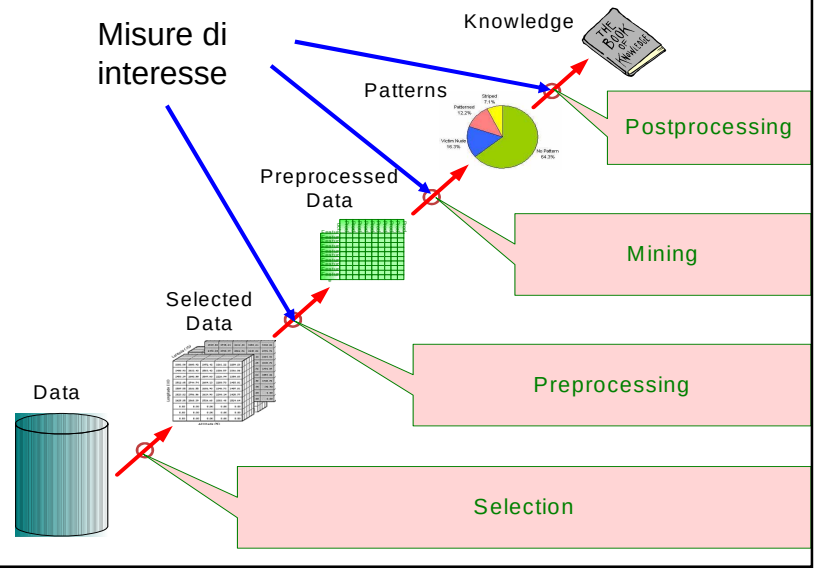
Valutazione delle regole

- Gli algoritmi per le regole associative tendono a produrre molte regole *inutili*
  - ✓ Molte di esse sono non interessanti (ovvie), altre possono essere ridondanti
    - C'è ridondanza se {A,B,C} → {D} e {A,B} → {D} hanno lo stesso supporto e confidenza
- L'utilizzo di **criteri/misure di interesse** possono permettere di eliminare/ordinare le regole associative fin qui costruite
- Si noti che sino a ora le uniche misure di interesse utilizzate sono il supporto e la confidenza

Misure di interesse

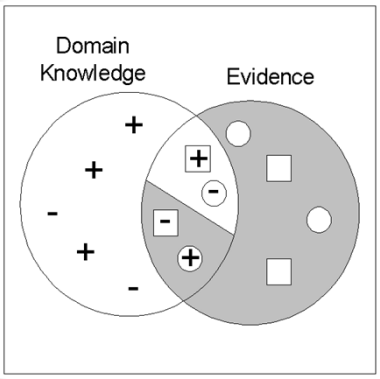
- **Misure oggettive:** danno priorità alle regole sulla base di criteri statistici calcolati a partire dai dati
  - ✓ Esistono molte formule a questo scopo, ognuna con i suoi pro e contro.
- **Misure soggettive:** danno priorità alle regole sulla base di criteri definiti dall'utente
  - ✓ Un pattern è interessante se contraddice le attese dell'utente
    - Un pattern è interessante se l'utente è interessato a svolgere qualche attività/ prendere qualche decisione relativamente agli elementi che lo compongono.
    - In questo caso si dice che il pattern è *actionable* (Silberschatz & Tuzhilin).Per esempio se sono intenzionato a fare una campagna promozionale sulla birra, sarò particolarmente interessato a tutte le regole associative che includono la birra

Quando applicare le misure di interesse?



Regole interessanti perchè inattese

- Richiedono di modellare le attese degli utenti ossia il dominio applicativo



+ Pattern che ci si attende siano frequenti

- Pattern che ci si attende siano non frequenti

Pattern rivelatesi frequenti

Pattern rivelatesi non frequenti

Pattern attesi

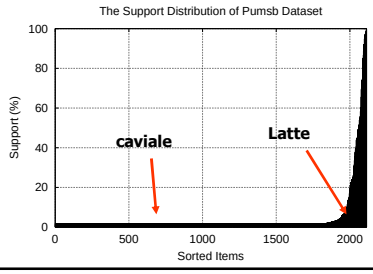
Pattern inattesi

- Non tutta la conoscenza sul domininio può essere catturata
- Difficile codificare la conoscenza sul dominio

Dataset con supporto non omogeneo

- Molti dataset presentano gruppi di item con supporto molto elevato assieme ad altri con supporto molto limitato
  - ✓ Una grande catena commerciale vende prodotti con range di prezzo da 1€ a 10.000€. Il numero di transazioni che includono prodotti con prezzo ridotto è di molto superiore a quelle con prezzo elevato. Tuttavia le associazioni tra questi ultimi sono di interesse per l'azienda.

Gruppi	G1	G2	G3
Supporto	<1%	1%≤ - ≤ 90%	> 90%
# item	1735	358	20



Dataset con supporto non omogeneo

- Fissare la soglia minsup per questi dataset può essere molto difficile
  - ✓ Una soglia troppo alta non permette di catturare le associazioni tra item con supporto limitato
  - ✓ Una soglia troppo bassa crea i seguenti problemi
    - Tempi di esecuzione elevati
    - Numero elevato di regole restituite
    - **Pattern cross-support**

Pattern cross-support

- Definizione: un pattern cross-support è un itemset $X=\{i_1,\dots,i_k\}$ per cui il rapporto dei supporti $r(X)$ è inferiore a una soglia data *thr*

$$r(X)=\frac{\min\{s(i_1),\dots,s(i_k)\}}{\max\{s(i_1),\dots,s(i_k)\}}$$

  - ✓ Consideriamo i seguenti item e il relativo supporto latte (70%), zucchero (10%) e caviale (0.04%). Se *thr*=0.01 l'itemset {latte, zucchero, caviale} è cross-support poiché 0.0004/0.7= 0.00058 < 0.01

- I pattern cross-support sono raramente interessanti e si preferisce studiare i pattern con valori di supporto "omogenei" e con alto valore di affinità tra **tutti** gli item che li compongono

Pattern cross-support

- Fissando la soglia *thr*=0.3 gli temset {*p*,*q*,*r*}, {*p*,*r*}, {*p*,*q*} risultano essere cross-support
- Le regole associative relative sono di scarso interesse anche se presentano confidenze elevate
  - ✓ c({*q*} → {*p*})=4/5=80%
- Sebbene tali pattern possano essere eliminati con un valore elevato per minsup (es. minsup=20%) il rischio è di perdere pattern che determinano regole di maggiore interesse
  - ✓ s({*p*,*q*})=4/30=13.3%
  - ✓ s({*q*,*r*})=5/30=16.7%
- In questa situazione supporto e confidenza non catturano adeguatamente la correlazione/affinità tra tutti gli elementi dell'itemset
  - ✓ c({*p*} →{*q*})=4/25=16% ha una confidenza molto bassa

p	q	r
0	1	1
1	1	1
1	1	1
1	1	1
1	1	1
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
1	0	0
0	0	0
0	0	0
0	0	0
0	0	0

H-confidence

- Dall'esempio precedente emerge che un pattern cross-support può essere individuato esaminando la confidenza più bassa che può essere estratta dall'itemset.
- Sfruttando la proprietà di anti-monotonicità della confidenza e dato un itemset $\{i_1,\ldots,i_k\}$, :

$$c(i_1 \rightarrow i_2..i_k) \leq c(i_1, i_2 \rightarrow i_3..i_k) \leq c(i_1, i_2, i_3 \rightarrow i_4,..i_k)$$

- In base a questa proprietà la regola con confidenza minore estraibile da un itemset frequente è quella che contiene un solo elemento nella parte sinistra
- Definizione: dato un itemset X=$\{i_1,\ldots,i_k\}$ il valore di h-confidence è definito da:

$$h-conf(X) = \min\{c(i_1 \rightarrow i_2..i_k), c(i_2 \rightarrow i_1, i_3,..i_k),..., c(i_k \rightarrow i_1,..i_{k-1})\}$$

- Sfruttando la proprietà di anti-monotonicità della confidenza è possibile dimostrare che h-conf(X) è un lower-bound a $r(X)$:

$$h\text{-}conf(X) \leq r(X)$$

H-confidence

- Dato un itemset frequente $X=\{i_1,\ldots,i_k\}$, la regola associativa

$$i_j \rightarrow i_1, i_2 i,..., _{j-1}, i_{j+1},..., i_k$$

è quella con confidenza minima se: $s(i_j) = \max\{s(i_1),..., s(i_k)\}$

- Quindi la confidenza minima riscontrabile in regole associative generate dall'itemset X è:

$$h-conf(X) = \frac{s(\{i_1,..., i_k\})}{\max\{s(i_1),..., s(i_k)\}}$$

ma per la proprietà di monotonicità del supporto

$$s(\{i_1,..., i_k\}) \leq \min\{s(i_1),..., s(i_k)\}$$

$$h-conf(X) \leq \frac{\min\{s(i_1),..., s(i_k)\}}{\max\{s(i_1),..., s(i_k)\}} = r(X)$$

- h-conf è inoltre monotona e quindi può essere sfruttata per il pruning negli algoritmi di mining

$$h-conf(\{i_1, i_2,..., i_k\}) \geq h-conf(\{i_1, i_2,..., i_k, i_{k+1}\})$$

Calcolo di misure di interesse

- Il caso dei pattern cross-support ha mostrato i limiti del supporto. Vediamo ora quali sono i limiti legati alla confidenza

- Premessa: data una regola $X \rightarrow Y$, le informazioni necessarie a calcolarne l'interesse possono essere ottenute dalla contingency table

Contingency table per $X \rightarrow Y$

	Y	$\overline{Y}$	
X	f_{11}	f_{10}	f_{1+}
$\overline{X}$	f_{01}	f_{00}	f_{o+}
	f_{+1}	f_{+0}	|T|

f_{11}: supporto per X e $\underline{Y}$
f_{10}: supporto per $\underline{X}$ e Y
f_{01}: supporto per $\underline{X}$ e $\underline{Y}$
f_{00}: supporto per X e Y

Limiti della Confidenza

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

- La regola Tea $\rightarrow$ Coffee ha supporto (15%) e confidenza (15/20=75%) elevati

Conf(Tea $\rightarrow$ Coffee) = P(Coffee|Tea) = P(Coffee $\wedge$ Tea)/ P(Tea)= 15/20=0.75

ciò sembra indicare che le persone che bevano Tea bevano anche Coffee.

- Eppure le persone che bevono Coffee (a prescindere dal fatto che bevano anche Tea) sono il 90%! Quindi la correlazione tra i due fatti è negativa!! E infatti:

Conf($\overline{\text{Tea}} \rightarrow$ Coffee) = P(Coffee|$\overline{\text{Tea}}$) = 0.75/0.80 = 0.9375 > P(Coffee) = 0.9

ossia se sappiamo che una persona non beve Tea è più probabile che beva Coffee.

Indipendenza statistica

- L'indipendenza stocastica di due eventi A e B si ha quando il verificarsi di uno non modifica la probabilità di verificarsi dell'altro, ovvero quando

$$P(A \mid B) = P(A) \qquad P(B \mid A) = P(B)$$

queste due condizioni si possono sintetizzare con la formula

$$P(A \wedge B) = P(A) \times P(B)$$

  - Se invece

$$P(A \wedge B) > P(A) \times P(B) \Rightarrow \text{Correlazione positiva}$$
$$P(A \wedge B) < P(A) \times P(B) \Rightarrow \text{Correlazione negativa}$$

  - Data una popolazione di 1000 studenti
  - ✓ 600 sanno nuotare (N)
  - ✓ 700 giocare a tennis (T)
  - ✓ 420 sanno nuotare e giocare a tennis (N,T)
  - ✓ $P(N \wedge T) = 420/1000 = 0.42 = P(N) \times P(T) = 0.6 \times 0.7$

Calcolo di misure di interesse

- Il limite della confidenza è dovuto al fatto che non considera il supporto dell'itemset nella parte destra della regola e quindi non fornisce una valutazione corretta nel caso in cui i gruppi di item non siano stocasticamente indipendenti
  - ✓ Si valuta che molte delle persone che bevono Tea bevono anche Coffee ma non si considera quante persone in totale bevono Coffee
- Una misura che tiene in considerazione questa eventualità è:

$$Lift(X \rightarrow Y) = \frac{c(X \rightarrow Y)}{s(Y)} = \frac{P(Y \mid X)}{P(Y)} = \frac{P(Y,X)}{P(X)}\frac{1}{P(Y)}$$

  - ✓ se lift=1 gli eventi sono indipendenti
  - ✓ se lift>1 gli eventi sono correlati positivamente
    - La probabilità di Y sapendo che si è verificato X è maggiore della probabilità di Y
  - ✓ se lift<1 gli eventi sono correlati negativamente
- Il lift indica come l'occorrenza di un evento fa salire le occorrenze dell'altro

Esempio: Lift

	Coffee	$\overline{\text{Coffee}}$	
Tea	15	5	20
$\overline{\text{Tea}}$	75	5	80
	90	10	100

c(Tea → Coffee)=75%

s(Coffee)=90%

lift(Tea → Coffee)=0.75/0.9=0.8333<1

Correlazione negativa! La regola non è interessante

Si supponga di aver 100 transazioni di vendita e 20 prodotti. Se il supporto per il prodotto *a* è 25%, il supporto per il prodotti *b* è 90% e il supporto per {*a, b*} è 20%. Siano poi le soglie di supporto e confidenza pari a 10% e 60%, Calcolare la confidenza della regola associativa {*a*} → {*b*}.
Calcolare la misura di interesse per il pattern {*a, b*} e descrivere la relazione tra *a* e *b*

Misure di interesse

- In letteratura sono state proposte molte misure di interesse
- Ogni misura è adatta ad alcune applicazioni ma non ad altre

#	Measure	Formula
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Goodman-Kruskal's (λ)	$\frac{\sum_j \max_k P(A_j,B_k)+\sum_k \max_j P(A_j,B_k)-\max_j P(A_j)-\max_k P(B_k)}{2-\max_j P(A_j)-\max_k P(B_k)}$
3	Odds ratio (α)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Q	$\frac{P(A,B)P(\overline{A}\overline{B})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{A}\overline{B})+P(A,\overline{B})P(\overline{A},B)} = \frac{\alpha-1}{\alpha+1}$
5	Yule's Y	$\frac{\sqrt{P(A,B)P(\overline{A}\overline{B})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{A}\overline{B})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}} = \frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
6	Kappa (κ)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
7	Mutual Information (M)	$\frac{\sum_i \sum_j P(A_i,B_j)\log\frac{P(A_i,B_j)}{P(A_i)P(B_j)}}{\min(-\sum_i P(A_i)\log P(A_i), -\sum_j P(B_j)\log P(B_j))}$
8	J-Measure (J)	$\max\left(P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})}),\right.$ $\left.P(A,B)\log(\frac{P(A\|B)}{P(A)})+P(\overline{A}B)\log(\frac{P(\overline{A}\|B)}{P(\overline{A})})\right)$
9	Gini index (G)	$\max\Big(P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A})[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]$ $-P(B)^2-P(\overline{B})^2,$ $P(B)[P(A\|B)^2+P(\overline{A}\|B)^2]+P(\overline{B})[P(A\|\overline{B})^2+P(\overline{A}\|\overline{B})^2]$ $-P(A)^2-P(\overline{A})^2\Big)$
10	Support (s)	$P(A,B)$
11	Confidence (c)	$\max(P(B\|A),P(A\|B))$
12	Laplace (L)	$\max\left(\frac{NP(A,B)+1}{NP(A)+2},\frac{NP(A,B)+1}{NP(B)+2}\right)$
13	Conviction (V)	$\max\left(\frac{P(A)P(\overline{B})}{P(A\overline{B})},\frac{P(B)P(\overline{A})}{P(B\overline{A})}\right)$
14	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
15	cosine (IS)	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
16	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
17	Certainty factor (F)	$\max\left(\frac{P(B\|A)-P(B)}{1-P(B)},\frac{P(A\|B)-P(A)}{1-P(A)}\right)$
18	Added Value (AV)	$\max(P(B\|A)-P(B),P(A\|B)-P(A))$
19	Collective strength (S)	$\frac{P(A,B)+P(\overline{A}\overline{B})}{P(A)P(B)+P(A)P(\overline{B})}\times\frac{1-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A,B)-P(\overline{A}\overline{B})}$
20	Jaccard (ζ)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
21	Klosgen (K)	$\sqrt{P(A,B)}\max(P(B\|A)-P(B),P(A\|B)-P(A))$

Misure di interesse

10 possibili contingency table

Example	f_{11}	f_{10}	f_{01}	f_{00}
E1	8123	83	424	1370
E2	8330	2	622	1046
E3	9481	94	127	298
E4	3954	3080	5	2961
E5	2886	1363	1320	4431
E6	1500	2000	500	6000
E7	4000	2000	1000	3000
E8	4000	2000	2000	2000
E9	1720	7121	5	1154
E10	61	2483	4	7452

Ordinamento delle contingency table in base alle diverse misure

#	ϕ	λ	α	Q	Y	κ	M	J	G	s	c	L	V	I	IS	PS	F	AV	S	ζ	K
E1	1	1	3	3	3	1	2	2	1	3	5	5	4	6	2	2	4	6	1	2	5
E2	2	2	1	1	1	2	1	3	2	2	1	1	1	8	3	5	1	8	2	3	6
E3	3	3	4	4	4	3	3	8	7	1	4	4	6	10	1	8	6	10	3	1	10
E4	4	7	2	2	2	5	4	1	3	6	2	2	2	4	4	1	2	3	4	5	1
E5	5	4	8	8	8	4	7	5	4	7	9	9	9	3	6	3	9	4	5	6	3
E6	6	6	7	7	7	7	6	4	6	9	8	8	7	2	8	6	7	2	7	8	2
E7	7	5	9	9	9	6	8	6	5	4	7	7	8	5	5	4	8	5	6	4	4
E8	8	9	10	10	10	8	10	10	8	4	10	10	10	9	7	7	10	9	8	7	9
E9	9	9	5	5	5	9	9	7	9	8	3	3	3	7	9	9	3	7	9	9	8
E10	10	8	6	6	6	10	5	9	10	10	6	6	5	1	10	10	5	1	10	10	7

Regole associative in presenza di attributi categorici e continui

- Fino a ora le variabili considerate sono di tipo binario e asimmetrico
  - ✓ Un item compare o non compare nella transazione
  - ✓ L'informazione rilevante è la sua presenza, non la sua assenza

Session Id	Country	Session Length (sec)	Number of Web Pages viewed	Gender	Browser Type	Buy
1	USA	982	8	Male	IE	No
2	China	811	10	Female	Netscape	No
3	USA	2125	45	Female	Mozilla	Yes
4	Germany	596	4	Male	IE	Yes
5	Australia	123	9	Male	Mozilla	No
…	…	…	…	…	…	…

- Come è possibile esprimere regole associative del tipo:

{Number of Pages $\in$ [5,10) $\wedge$ (Browser=Mozilla)} $\rightarrow$ {Buy = No}

Gestione degli attributi categorici

- La soluzione proposta è quella di trasformare gli attributi categorici in attributi binari asimmetrici introducendo un nuovo "item" per ogni possibile valore dell'attributo
  - ✓ L'attributo Browser type sarà sostituito da:
    - Browser Type = Internet Explorer
    - Browser Type = Mozilla
    - Browser Type = Netscape

Session Id	Country	Session Length (sec)	Number of Web Pages viewed	Male	Female	Internet explorer	…	Buy
1	USA	982	8	Yes	No	Yes	…	No
2	China	811	10	No	Yes	No	…	No
3	USA	2125	45	No	Yes	No	…	Yes
4	Germany	596	4	Yes	No	Yes	…	Yes
5	Australia	123	9	Yes	No	No	…	No
…	…	…	…	…	…	…	…	…

Gestione degli attributi categorici

- Attenzione ai casi in cui l'attributo categorico abbia un numero molto elevato di possibili valori: per alcuni di questi valori/attributi sarà difficile raggiungere la soglia di supporto minimo
  - ✓ Abbassare la soglia di supporto minimo può causare un forte aumento dei tempi di calcolo oltre a determinare molti pattern non interessanti
  - ✓ E' più utile aggregare tra loro più gruppi di valori dell'attributo
- Nel caso in cui un attributo abbia una distribuzione delle istanze dei valori molto disomogenea (es. La maggioranza degli utenti utilizzi Mozzilla) saranno generati molti pattern ridondanti
  - ✓ La regola $\{\text{Number of Pages} \in [5,10) \wedge (\text{Browser=Mozilla})\} \rightarrow \{\text{Buy = No}\}$ è sussunta dalla più generale regola

    $\{\text{Number of Pages} \in [5,10)\} \rightarrow \{\text{Buy = No}\}$
- In questo caso è preferibile eliminare l'attributo binario corrispondente al valore con elevato supporto poiché esso non fornisce informazioni utili
- Una soluzione alternativa è quella di utilizzare la tecnica già vista per gestire attributi con supporto disomogeneo

Gestione degli attributi categorici

- In ogni caso la trasformazione degli attributi categorici comporta un aumento del numero di attributi considerati e di conseguenza un aumento del tempo di calcolo delle soluzioni.
- Una tecnica per ridurre questo overhead è di evitare la generazione di itemset candidati che includano più item provenienti dallo stesso attributi poichè il conteggio del supporto dell'itemset sarà certamente 0.
  - ✓ Supp({State='Italia', State='Germania'}) =0

Gestione degli attributi continui

- Le regole associative che includono attributi a valori continui sono dette regole associative quantitative:
  - ✓ Age$\in$[21,35) $\wedge$ Salary$\in$[70k,120k) $\rightarrow$ Buy
- Per la loro gestione possono essere utilizzate più approcci:
  - ✓ Basati sulla discretizzazione
  - ✓ Basati sulla statistica
  - ✓ Senza discretizzazione
    - minApriori

Gestione degli attributi continui

- La discretizzazione pone il problema di come fissare il numero e il confine degli intervalli
- Il numero degli intervalli è normalmente fornito dagli utenti e può essere espresso in termini di:
  - ✓ Ampiezza degli intervalli: discretizzazione equi-width
  - ✓ Numero medio di transazioni per intervallo: discretizzazione equi-depth (o equi-frequency)
  - ✓ Numero di cluster
- La scelta dell'ampiezza degli intervalli incide sul valore di supporto e confidenza:
  - ✓ Intervalli troppo ampi riducono il valore della confidenza
  - ✓ Intervalli troppo ridotti riducono il valore del supporto, e tendono a determinare regole replicate

Gestione degli attributi continui

Age group	Chat online= Y	Chat online = N
[12,16)	12	13
[16,20)	11	2
[20,24)	11	3
[24,28)	12	13
[28,32)	14	12
[32,36)	15	12
[36,40)	16	14
[40,44)	16	14
[44,48)	4	10
[48,52)	5	11
[52,56)	5	10
[56,60)	4	11
TOTAL	**125**	**125**

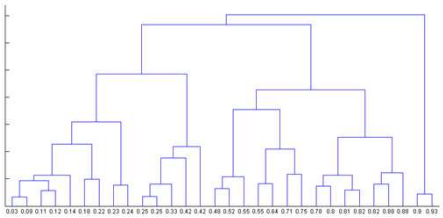
- Questo dataset è caratterizzato da due pattern
  - ✓ R1: Age $\in$ [16,24)$\rightarrow$ Chat online =Y (s=8.8% C=81.5%)
  - ✓ R2: Age $\in$ [44,60)$\rightarrow$ Chat online =N (s=16.8% C=81.5%)

Gestione degli attributi continui

- Adottando valori di soglia minsupp=5% minconf=65%
- Utilizzando intervalli ampi si riduce la confidenza dei pattern poichè il raggruppamento non cattura più un fenomeno omogeneo
  - ✓ R1: Age ∈ [12,36)→ Chat online =Y (s=30% C=57.7%)
  - ✓ R2: Age ∈ [36,60)→ Chat online =N (s=28% C=58.3%)
- Utilizzando intervalli stretti si riduce il supporto dei pattern poichè al raggruppamento corrispondono un minor numero di transazioni
  - ✓ R1: Age ∈ [16,20)→ Chat online =Y (s=4.4% C=70%)
  - ✓ R2: Age ∈ [20,24)→ Chat online =N (s=4.4% C=70%)
- Utilizzando intervalli di ampiezza 8 la regola 2 determina 2 sotto-regole
  - ✓ R21: Age ∈ [44,52)→ Chat online =N (s=8.4% C=70%)
  - ✓ R22: Age ∈ [52,60)→ Chat online =N (s=8.4% C=70%)

Gestione degli attributi continui

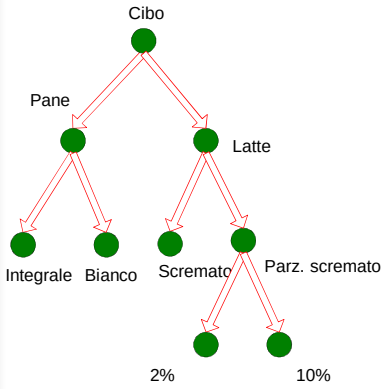
- Una possibile soluzione è quella di provare con tutti i possibili intervalli
  - ✓ Si utilizza inizialmente un range di larghezza k, intervalli vicini sono poi fusi progressivamente.

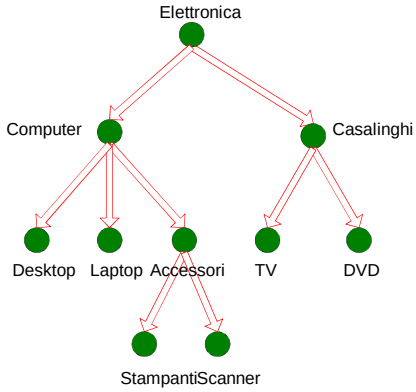


  - ✓ Il tempo di esecuzione può diventare proibitivo
  - ✓ La soluzione porta alla generazione di molte regole ridondanti
    - {Refund = No, (Income = $51,250)} → {Cheat = No}
    - {Refund = No, (51K ≤ Income ≤ 52K)} → {Cheat = No}
    - {Refund = No, (50K ≤ Income ≤ 60K)} → {Cheat = No}

Regole associative multi-livello

- Una gerarchia di concetti è: una gerarchia di generalizzazione basata sulla semantica dei suoi elementi





Regole associative multi-livello

- Perchè incorporare gerarchie di concetti?
  - ✓ Le regole ai livelli più bassi di una gerarchia potrebbero non avere supporto sufficiente per apparire in itemset frequenti
  - ✓ Le regole ai livelli inferiori potrebbero essere troppo specifiche
    - Latte scremato → pane bianco
    - Latte parzialmente scremato→ pane bianco,
    - Latte scremato → pane integrale

Sono tutte indicative di un'associazione più generale tra latte e pane

Regole associative multi-livello

- Le regole associative multi-livello possono essere gestite con gli algoritmi già studiati estendendo ogni transazione con gli item genitori degli item presenti nella transazione…
  - ✓ Transazione originale: {latte scremato, pane bianco}
  - ✓ Transazione estesa: {latte scremato, pane bianco, latte, pane, cibo}
- .. ridefinendo il concetto di associazione al fine di evitare la scoperta di associazioni inultili
  - ✓ $X \Rightarrow Y$
  - ✓ $X \cap Y = \{\}$ Y non deve contenere nessun antenato di nessuno degli elementi di X

Regole associative multi-livello

- Conseguenze:
  - ✓ Gli item ai livelli "alti" della gerarchia avranno supporti molto elevati
    - Possibilità di pattern cross-support
    - Se la soglia per il supporto è bassa saranno identificati molti pattern che coinvolgono item ai livelli alti della gerarchia
    - Il numero di associazioni ridondanti aumenta
      - – Latte $\Rightarrow$ Pane
      - – Latte scremato $\Rightarrow$ Pane
  - ✓ La dimensionalità dei dati aumenta e conseguentemente aumenta il tempo di elaborazione

Regole associative multi-livello

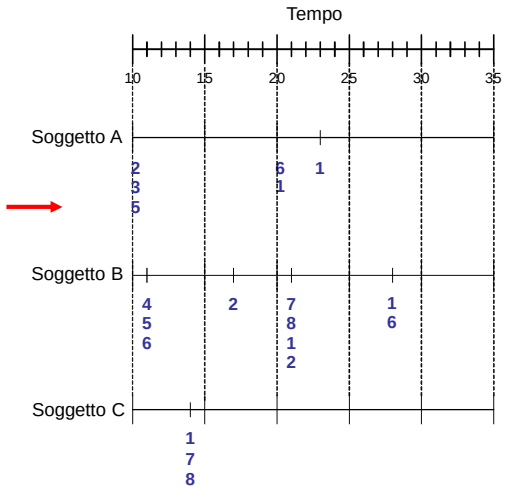
- Una soluzione alternativa è quella di generare i pattern frequenti separatamente per i diversi livelli della gerarchia
  - ✓ Generare inizialmente i pattern al livello più alto della gerarchia
  - ✓ Procedere iterativamente verso i livelli successivi
- Conseguenze
  - ✓ Il costo di I/O crescerà notevolmente poichè saranno necessarie più scansioni dei dati
  - ✓ Saranno persi eventuali associazioni cross-livello interessanti

Pattern sequenziali

- Spesso alle transazioni sono associate informazioni temporali che permettono di collegare tra loro gli eventi che riguardano uno specifico soggetto

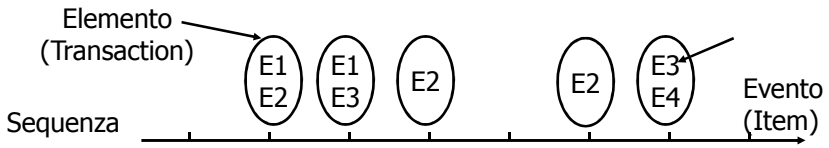
Sequence Database:

Soggetto	Tempo	Eventi
A	10	2, 3, 5
A	20	6, 1
A	23	1
B	11	4, 5, 6
B	17	2
B	21	7, 8, 1, 2
B	28	1, 6
C	14	1, 8, 7



Dati sequenziali: alcuni esempi

Database	Sequenza	Elemento (Transaction)	Evento (Item)
Clienti	Storia degli acquisti di un cliente	L'insieme degli item comprati da un cliente al tempo t	Libri,CD, ecc.
Dati web	Attività di browsing di un particolare visitatore web	Una collezione di file visualizzati da un visitatore web dopo un singolo click del mouse	Home page, index page, contact info, ecc.
Eventi	Storia degli eventi generati da un sensore	Eventi scatenati dal sensore al tempo t	Tipi di allarmi generati dal sensore
Sequenze genomiche	Sequenze del DNA di una particolare specie	Un lemento della sequenza del DNA	Basi A,T,G,C



Definizione di sequenza

- Una sequenza è una lista ordinata di elementi (transazioni)

  $s = < e_1 e_2 e_3 \ldots >$

  - ✓ Ogni elemento contiene un insieme di eventi (item)

    $e_i = \{i_1, i_2, \ldots, i_k\}$

  - ✓ A ogni elemento è associato uno specifico istante temporale o posizione ordinale

- La lunghezza della sequenza, |s|, è data dal numero degli elementi che la compongono
- Mentre una k-sequence è una sequenza che contiene k eventi
- ATTENZIONE le sequenze formate da k eventi possono avere lunghezze diverse

  < {1,2,3} > < {1,2} {3} > < {1} {2} {3} >

Definizione di sottosequenza

- Una sequenza $<a_1\ a_2 \ldots a_n>$ è contenuta in una sequenza $<b_1\ b_2 \ldots b_m>$ $(m \geq n)$ se esistono degli interi $i_1 < i_2 < \ldots < i_n$ tali che $a_1 \subseteq b_{i_1}$, $a_2 \subseteq b_{i_2}$, …, $a_n \subseteq b_{i_n}$

Sequenze	Sottosequenze	E' contenuta?
< {2,4} {3,5,6} {8} >	< {2} {3,5} >	Si (1,2)
< {1,2} {3,4} >	< {1} {2} >	No
< {2,4} {2,4} {2,5} >	< {2} {4} >	Si (1,2)

- Il supporto di una sottosequenza w è definito come la frazione di sequenze che contengono w
- Un *pattern sequenziale* è una sottosequenza frequente ossia il cui supporto è ≥ *minsup*

Mining di pattern sequenziali

- Dato un database di sequenze e una soglia di supporto minimo, *minsup* trovare tutte le sottosequenze il cui supporto sia ≥ *minsup*

Database di sequenze

SID	sequence
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>
40	<eg(af)cbc>

<a(bc)dc> è una *sottosequenza* di <a(abc)(ac)d(cf)>

Data *una soglia minsup* =2, <(ab)c> è un *pattern sequenziale*

- La ricerca di pattern sequenziali è un problema difficile visto il numero esponenziale di sottosequenze contenute in una sequenza
  - ✓ Il numero di k-sottosequenze contenute in una sequenza con n eventi è: $\binom{n}{k}$
  - ✓ Una sequenza con 9 elementi contiene: $\binom{9}{1}+\binom{9}{2}+\ldots+\binom{9}{9}=2^9-1=516$ sequenze

Tecniche per il mining di pattern sequenziali

- Approcci basati sul principio Apriori
  - ✓ **GSP (implementato in Weka)**
  - ✓ SPADE
- Approcci basati sul principio Pattern-Growth
  - ✓ FreeSpan
  - ✓ PrefixSpan

Approccio naive

- Dati n eventi: i_1, i_2, i_3, …, i_n, enumerare tutte le possibili sequenze e calcolare il relativo supporto
  - ✓ 1-sottosequenze candidate:
    $<\{i_1\}>, <\{i_2\}>, <\{i_3\}>, \ldots, <\{i_n\}>$
  - ✓ 2-sottosequenze candidate:
    $<\{i_1, i_2\}>, <\{i_1, i_3\}>, \ldots, <\{i_1\}\ \{i_1\}>, <\{i_1\}\ \{i_2\}>, \ldots, <\{i_{n-1}\}\ \{i_n\}>$
  - ✓ 3-sottosequenze candidate:
    $<\{i_1, i_2, i_3\}>, <\{i_1, i_2, i_4\}>, \ldots, <\{i_1, i_2\}\ \{i_1\}>, <\{i_1, i_2\}\ \{i_2\}>, \ldots,$
    $<\{i_1\}\ \{i_1, i_2\}>, <\{i_1\}\ \{i_1, i_3\}>, \ldots, <\{i_1\}\ \{i_1\}\ \{i_1\}>, <\{i_1\}\ \{i_1\}\ \{i_2\}>, \ldots$
- Si noti che rispetto alle regole associative il numero di sottosequenze candidate è di molto superiore al numero degli itemset candidati poiché:
  - ✓ Un item può apparire una sola volta, ma un evento può apparire più volte, poiché nelle sequenze conta l'ordinamento
    $<\{i_1, i_2\}>, <\{i_1\}\ \{i_2\}>, <\{i_2\}\ \{i_1\}>$

Principio Apriori e algoritmo GSP

- Il principio Apriori si può applicare anche nel caso di pattern sequenziali poiché:
  - ✓ qualsiasi sequenza che contenga una particolare k-sequenza *s* deve contenere tutte le (k-1)-sottosequenze di *s*

```
k=1
F_k = { i | i ∈ I ∧ σ({i}) ≥ N × minsupp}
// trova le 1-sequence frequenti
repeat
  k=k+1
  C_k=apriori-gen(F_k-1) // genera le k-subsequence candidate
   for each sequence t ∈ T
    C_t = subsequence(C_k,t)
    // determina le sottosequenze candidate che compaiono in t
    for each candidate k-subsequce c ∈ C_t
     σ(c) = σ(c)+1 // incrementa il supporto
    end for
  end for
  F_k = {c | c ∈ C_k ∧ σ(c) ≥ N × minsupp}
  // identifica le k-sequenze frequenti
until F_k =∅
Risultato = ∪F_k
```

Algoritmo GSP: Generalized Sequential Pattern

- **Step 1**:
  - ✓ Fai una prima scansione del DB delle sequenze per individuare tutte le 1-sequenze
- **Step 2**:

  Ripeti fino a che sono scoperte nuove sequenze frequenti
  - ✓ **Generazione dei candidati**:
    - Fondi coppie di sottoseqenze frequenti trovate al passo *k*-1 per generare sequenze candidate che contengono *k* item
  - ✓ **Pruning dei candidati**:
    - Eliminalle *k*-sequenze candidate che contengono (*k-1)*-sottosequenze non frequenti
  - ✓ **Conteggio del supporto**:
    - Fai una scansione del DB per trovare il supporto delle sequenze candidate
  - ✓ **Eliminazione dei candidati**:
    - Elimina le *k*-sequenze candidate il cui supporto è effettivamente inferiore a *minsup*

Generazione dei candidati

- Caso base (k=2):
  - ✓ La fusione di due 1-sequenze frequenti $<\{i_1\}>$ e $<\{i_2\}>$ produrrà due 2-sequenze candidate: $<\{i_1\}\ \{i_2\}>$ e $<\{i_1, i_2\}>$
- Caso generale (k>2):
  - ✓ Una *(k-1)*-sequenza frequente w_1 è fusa con un'altra *(k-1)*-sequenza frequente w_2 per produrre una *k*-sequenza candidata se rimuovendo il primo evento in w_1 e rimuovendo l'ultimo evento in w_2 si ottiene la stessa sottosequenza
  - ✓ La k-sequenza ottenuta corrisponde a w_1 estesa con l'ultimo evento in w_2.
    - Se gli ultimi due eventi in w_2 appartengono allo stesso elemento, allora l'ultimo evento in w_2 diventa parte dell'ultimo elemento in w_1
    - Altrimenti, l'ultimo elemento in w_2 diventa un elemento separato aggiunto alla fine di w_1

GSP: un esempio

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

→ Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

→ Candidate Pruning

< {1} {2 5} {3} >

- La fusione delle sequenze w_1=<{1} {2} {3}> e w_4 =<{2} {3} {4}> produce la sequenza candidata < {1} {2} {3} {4}> dato che gli eventi {3} e {4} appartengono a elementi separati in w_4

GSP: un esempio

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

Candidate Pruning

< {1} {2 5} {3} >

- Le sequence w_1 =<{1} {2} {3}> e w_2 =<{1} {2,5}> non devono essere fuse poichè rimuovendo il primo elemento da w_1 e l'ultimo da w_2 non si ottiene la medesima sotto sequenza (<{2} {3}> ≠< {2,5}>)
- <{1} {2,5} {3}> è un candidato generato fondendo <{1} {2,5} > e <{2,5} {3}> poichè <{1} **{2,5}** > = <**{2,5}** {3}>

GSP: un esempio

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

Candidate Pruning

< {1} {2 5} {3} >

- La fusione delle sequenze w_3=<{1} {5} {3}> e w_7 =<{5} {3,4}> produce la sequenza candidata < {1} {5} {3,4}> dato che gli eventi {3} e {4} appartengono allo stesso elemento in w_7

Vincoli temporali

- La ricerca di pattern sequenziali significativi può essere resa più efficace imponendo vincoli temporali sulla struttura delle sequenze:

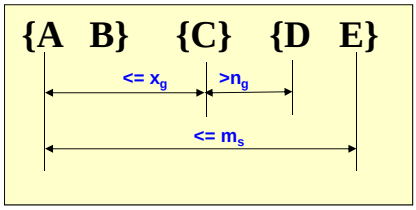
  Studente A < {CPS} {Basi di dati} {Data mining} >

  Studente b < {Basi di dati} {CPS} {Data mining} >

  - ✓ Entambe gli studenti rispondono al requisito in base al quale per poter seguire l'esame di data mining è necessario avere sostenuto gli esami di Basi di dati e Calcolo delle probabilità
  - ✓ Tuttavia i pattern non esprimono il vincolo per cui tali esami non possono essere sostenuti 10 anni prima poiché l'intervallo temporale sarebbe troppo elevato

Vincoli temporali

- **MaxSpan:** specifica il massimo intervallo temporale tra il primo e l'ultimo evento nella sequenza
  - ✓ Aumentando MaxSpan aumenta la probabilità di trovare una sottosequenza in una sequenza ma aumenta anche il rischio di correllare due eventi troppo distanti temporalmente
- **MinGap:** specifica il minimo intervallo temporale che deve trascorrere tra il verificarsi di eventi contenuti in due elementi diversi
- **MaxGap:** specifica il massimo intervallo temporale entro il quale gli eventi contenuti in un elemento devono svolgersi rispetto a quelli contenuti nell'evento precedente



x_g: MaxGap

n_g: MinGap

m_s: MaxSpan

Vincoli temporali: un esempio

- Assumendo che gli elementi siano eseguiti in istanti successivi, si valuti se le seguenti sottosequenze soddisfano i seguenti vincoli temporali
  - ✓ MaxSpan=4
  - ✓ MinGap=1
  - ✓ MaxGap=2

Sequenze	Sottosequenze	Soddisfa?
< {2,4} {3,5,**6**} {4,7} {4,**5**} {8} >	< {6} {5} >	SI
< {**1**} {2} {3} {**4**} {5}>	< {1} {4} >	No MaxGap
< {1} {**2**,3} {**3**,4} {4,**5**}>	< {2} {3} {5} >	SI
< {**1**,**2**} {3} {2,3} {3,4} {2,4} {4,**5**}>	< {1,2} {5} >	No MaxSpan+MaxGap

Mining di pattern sequenziali con vincoli temporali

- I vincoli precedenti incidono sul supporto dei pattern riducendolo
  - ✓ Alcuni pattern conteggiati come frequenti potrebbero non esserlo poichè alcune delle sequenze nel loro supporto potrebbero violare un vincolo temporale
  - ✓ E' necessario modificare le tecniche di conteggio per tenere conto di questo problema
- Sono possibili due soluzioni
  - ✓ Approccio 1
    - Calcolare le sottosequenze frequenti senza considerare i vincoli temporali
    - Applicare i vincoli temporali a posteriori
  - ✓ Approccio 2
    - Modificare GSP per eliminare direttamente i candidati che violano i vincoli temporali
    - ATTENZIONE questa soluzione può portare alla violazione del principio APriori per il vincolo MaxGap

Mining di pattern sequenziali con vincoli temporali

Soggetto	Timestamp	Eventi
A	1	1,2,4
A	2	2,3
A	3	5
B	1	1,2
B	2	2,3,4
C	1	1, 2
C	2	2,3,4
C	3	2,4,5
D	1	2
D	2	3, 4
D	3	4, 5
E	1	1, 3
E	2	2, 4, 5

Supponiamo che:

x_g = 1 (max-gap)

n_g = 1 (min-gap)

m_s = 5 (maximum span)

minsup = 60%

<{2} {5}> supporto = 40%

ma

<{2} {3} {5}> supporto = 60%

Esplicitare le sequenze contenute nelle transazioni

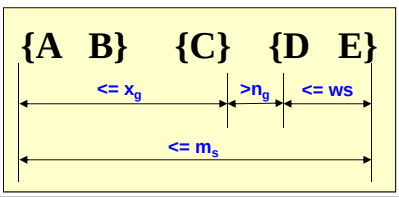
Il problema nasce dalla violazione del vincolo MaxGap che è invece soddisfatto se si inserisce l'elemento {3} che riduce i tempi tra elementi successivi

Vincoli temporali

- Un ulteriore tipo di vincolo temporale che però rilassa la definizione di base è quello di **Time Window Size** (ws) ossia l'intervallo temporale entro il quale due eventi avvenuti in tempi diversi devono essere considerati *contemporanei*

- Dato un pattern candidato <{a, c}> qualsiasi sequenza che contenga:
  - ✓ <… {a c} … >,
  - ✓ <… {a} … {c}…> (con time({c}) – time({a}) ≤ ws)
  - ✓ <…{c} … {a} …> (con time({a}) – time({c}) ≤ ws)

  contribuisce al supporto del pattern candidato



x_g: max-gap

n_g: min-gap

ws: window size

m_s: maximum span

Vincoli temporali: un esempio

- Assumendo che gli elementi siano eseguiti in istanti successivi, si valuti se le seguenti sottosequenze soddisfano i seguenti vincoli temporali
  - ✓ MaxSpan=5
  - ✓ MinGap=1
  - ✓ MaxGap=2
  - ✓ WindowSize=1

Sequenze	Sottosequenze	Soddisfa?
< {2,4} {3,5,6} {4,7} {4,6} {8} >	< {3} {5} >	No
< {1} {2} {3} {4} {5}>	< {1,2} {3} >	Si
< {1,2} {2,3} {3,4} {4,5}>	< {1,2} {3,4} >	Si