Lesson 5

Stream Frequent Itemsets and Association Rule Mining

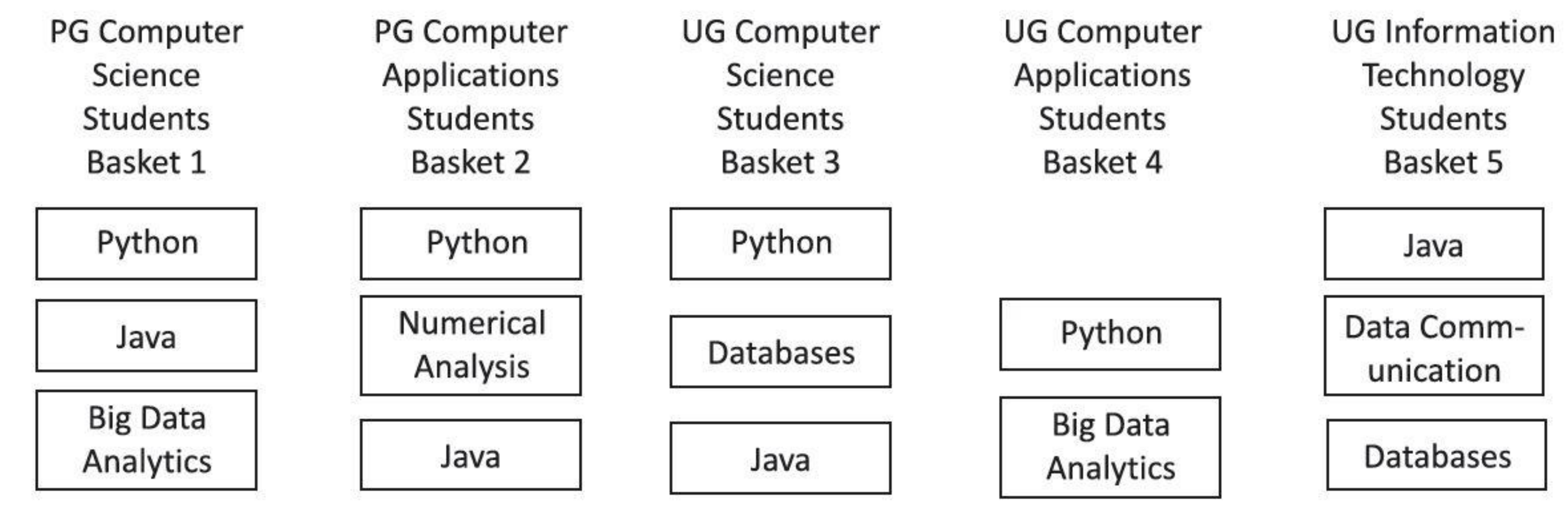
Frequent Itemset in a stream

- Frequent Itemset (FI) refers to a subset of items that appears frequently in the streaming datasets
- Refers to a set of items that frequently appear together
- .

Frequent Itemset Mining (FIM)

- Frequent itemsets and frequent patterns have many applications
- For example, Python and Big Data Analytics when the students of computer science frequently chose these subjects for in-depth studies

Figure 7.6 Courses organization



PG Computer Science Students Basket 1	PG Computer Applications Students Basket 2	UG Computer Science Students Basket 3	UG Computer Applications Students Basket 4	UG Information Technology Students Basket 5
Python	Python	Python		Java
Java	Numerical Analysis	Databases	Python	Data Comm-unication
Big Data Analytics	Java	Java	Big Data Analytics	Databases

Frequent substructure Mining (FIM)

- Refers to finding different structural forms, such as graphs, trees or lattices, which may be combined with itemsets or subsequences
- Provides the knowledge of important pairs of items that occur much more frequently than the items bought independently

FIM Algorithm

- A technique to extract knowledge from data
- Extracts on frequently occurring entities, events, …
- Finds the regularities in data

FIM Algorithm

- Specifies a given minimum frequency threshold for considering an itemset as frequent
- The extraction generally depends on the specified threshold

FIM Algorithm

- Is preceding step to the association rule learning (mining) algorithm
- For example, customers of supermarkets, mail order companies and online shops use FIM to find a set of products that are frequently bought together (association)

Association Rule Mining

- The goal of association rule mining is to discover items that are found together in sufficient number of baskets
- To find dependencies among these items
- This simply implies finding frequent itemsets

Apriori principle

- Apriori Algorithm (Section 6.5.3): uses iterations (successive passes).
- Algorithm Apriori limits the need for the main memory.
- The first pass needs memory proportional to the number of items. The second pass needs memory proportional to the square of frequent items only (for counts).

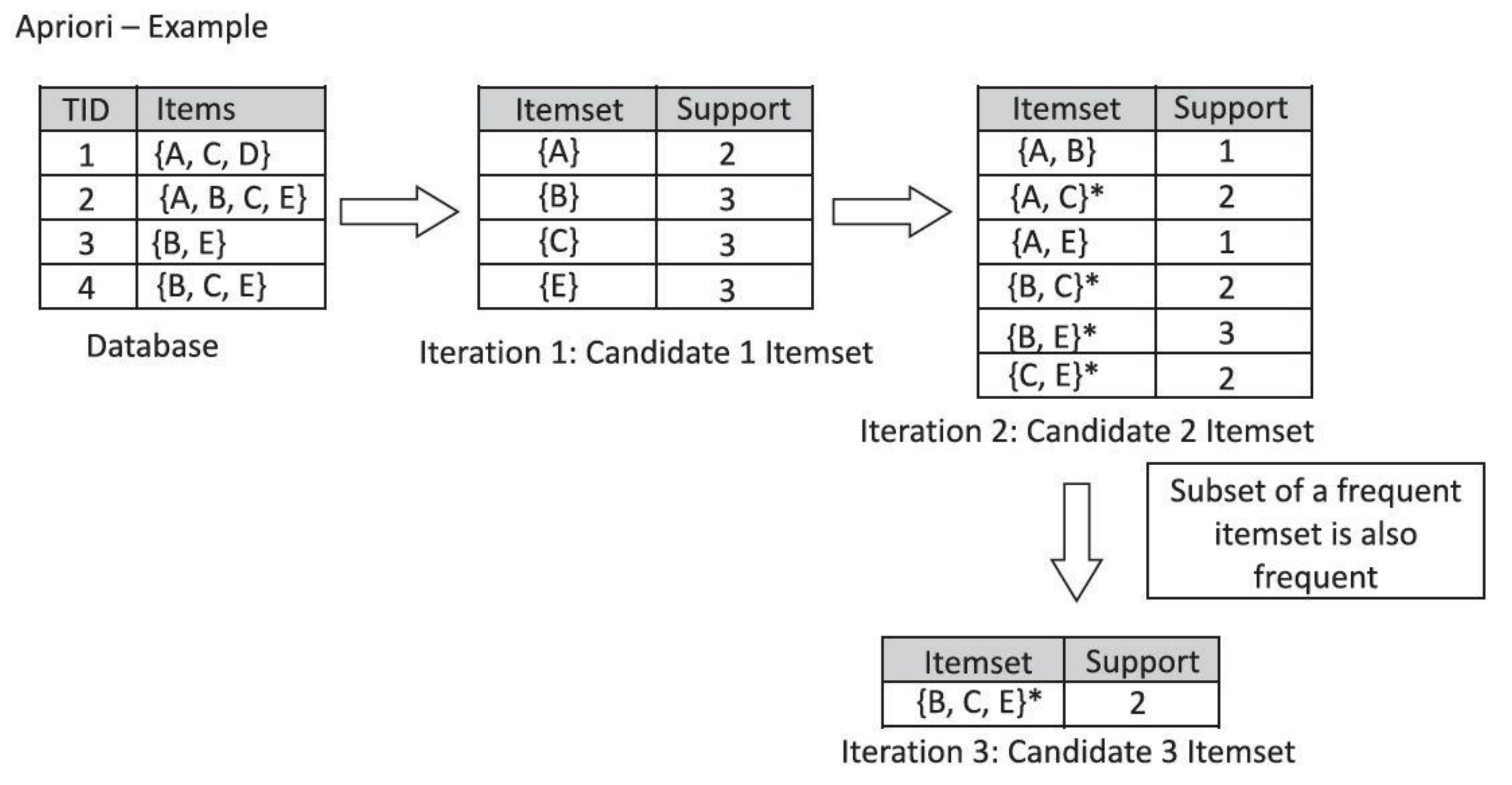
… Apriori principle

- Several proposed algorithms cut down on the size of candidate pairs. Suggests if an itemset is frequent, then all of its subsets must also be frequent
- For example, if itemset {A, B, C} is a frequent itemset, then all of its subsets {A}, {B}, {C}, {A, B}, {B, C} and {A, C} must be frequent.

… Apriori principle

- On the contrary, if an itemset is not frequent, then none of its supersets can be frequent. (Superset means a set consisting of the members which includes the itemsets in the subsets)
- This results into a smaller list of potential frequent itemsets (FIs) as the mining progresses.

Figure 6.8:
Apriori algorithm process for adopting the subset of frequent itemsets as a frequent itemset.



Apriori – Example

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

Database

Itemset	Support
{A}	2
{B}	3
{C}	3
{E}	3

Iteration 1: Candidate 1 Itemset

Itemset	Support
{A, B}	1
{A, C}*	2
{A, E}	1
{B, C}*	2
{B, E}*	3
{C, E}*	2

Iteration 2: Candidate 2 Itemset

Subset of a frequent itemset is also frequent

Itemset	Support
{B, C, E}*	2

Iteration 3: Candidate 3 Itemset

Formal statement of the association rule finding problem

- Let $\mathscr{I} = \{I1, I2, \ldots, Id\}$ be a set of d distinct attributes, also called literals
- Let $\mathfrak{I} = \{t1, t2, \ldots, tn\}$ be set of n transactions and contain a set of items such that $\mathfrak{I} \subseteq \mathscr{I}$
- $\subseteq$ means a subset of and $\subset$ means proper (strict) subset

Formal statement of the association rule finding problem

- An association rule is an implication of the form, X → Y, where X, Y belong to sets of items called itemsets ($X, Y \subset I$), and X and Y are disjoint itemsets ($X \cap Y = \emptyset$).

Explanation

- $\cap$ means intersection
- ⦰ means disjoint (no common members).
- Here, X is called antecedent, and Y consequent.

Association Rule Form

- if ( ) then ( ) form (Condition)
- ‘If’ part is called antecedent
- ‘Then’ part is called consequent (Result)

Association Rule Form

- If-then rules about the contents of baskets: $\{p1, p2 \ldots, pk\} \rightarrow q$ *means,*
- "If a basket contains all of $p1, p2 \ldots, pk$ then it is likely to contain q."

Algorithm of Park, Chen and Yu (PCY)

- Takes benefit of the fact that the first iteration of Apriori does not use lots of main memory for counting of single items
- Iteration 1 of PCY algorithm saves item counts as well as maintains a hash table with sufficient buckets that fits in memory

Algorithm of Park, Chen and Yu (PCY)

- PCY also maintains the counts for each bucket into which pairs of items are hashed.
- An improved PCY exists: Between the iterations the buckets are replaced by a bit vector

Multistage Algorithm

- A refinement of the PCY algorithm is the multistage algorithm
- The algorithm uses several successive hash tables to reduce the number of candidate pairs subsequently
- The algorithm applies more than two iterations to find the frequent pairs.

Multistage Algorithm

- The idea is to rehash only those pairs that qualify for iteration 2 of PCY after iteration 1 of PCY
- Since only a few pairs contribute to buckets in the middle iteration, fewer false positives may occur
- Requires 3 iterations, two hash functions have to be independent

Multihash Algorithm

- An improvement of PCY
- Main idea is to use several independent hash tables during the first iteration. This can lead to benefits like multistage in only 2 iterations.

Savasere, Omiecinski and Navathe **(SON) Algorithm**

- Keeps away from both false negatives and false positives using two passes
- Algorithm repetitively reads small subsets of the baskets into the main memory, and run an in-memory algorithm to find all the frequent itemsets.

SON Algorithm

- Subsets are not samples. It is the processing of the entire file in memory-sized chunks
- An itemset becomes a candidate if it is found to be frequent in any one or more subsets of the baskets.

SON Algorithm

- Second pass counts all the candidate itemsets and determine those which are frequent in the entire set
- The idea of monotonicity used here is that an itemset cannot be frequent in the entire set of baskets unless it is frequent in at least one subset.

SON Distributed Environment Algorithm

- Implements in a parallel computing environment
- The implementation distributes the baskets among many nodes
- Frequent itemsets compute at multiple nodes. Finally, accumulate the counts of all the candidates.

Toivonen Algorithm

- Similar to the simple random sample algorithm but lowers the threshold slightly for sampling
- For example, if the sample s is 1% of the baskets, use 0.008 s as the support threshold rather than 0.01 s, so as is not to miss any frequent itemset in the full set of baskets

Frequent Itemsets in Decaying Windows

- The decaying window method for identifying the most common elements in a stream
- The weight of ith previous item assigns as $(1 – C)^i \approx e^{–ci}$ where $0 < (1 – C)^i \leq 1$ where $i \geq 1$. x

Decaying Windows Modification

- Counting frequent items requires two modifications to the algorithm

(1) Stream elements are baskets, and not the individual items. Maintain a weighted count for itemsets. When a new itemset arrives,

Decaying Windows Modification

(i) Multiply all previous counts by 1 – C.

(ii) Add a new itemset with an initial count of 1.

(iii) Add 1 to an existing itemsets count.

Decaying Windows Modification

(2) Start counting an itemset only if all of its proper subsets are already being counted (Remember from the Apriori algorithm that if an itemset is frequent, then all of its subsets must also be frequent).

Summary

We learnt:

- Frequent Itemsets Mining
- Apriori Algorithm
- Association Rule
- Park, Chen and Yu
- Multihash algorithm

Summary

We learnt:

- SON Algorithm
- Toivonen's Algorithm
- Counting Frequent Items in a Stream
- Frequent Itemsets using Decaying Windows

End of Lesson 5 on

Stream Frequent Itemsets and Association Rule Mining