Association Pattern Mining - Part 2

But…

- Indeed, Apriori significantly reduces the size of candidate sets, leading to good performance gain.
- However, it can suffer from a nontrivial costs:
  - *It may still need to generate a huge number of candidate sets.* For example, if there are 10^4 frequent 1-itemsets, the Apriori algorithm will need to generate more than 10^7 candidate 2-itemsets.
- *Can we design a method that mines the complete set of frequent itemsets without such a costly candidate generation process?*

FP-growth (finding frequent itemsets without candidate generation)

Frequent Pattern Growth is a method that:

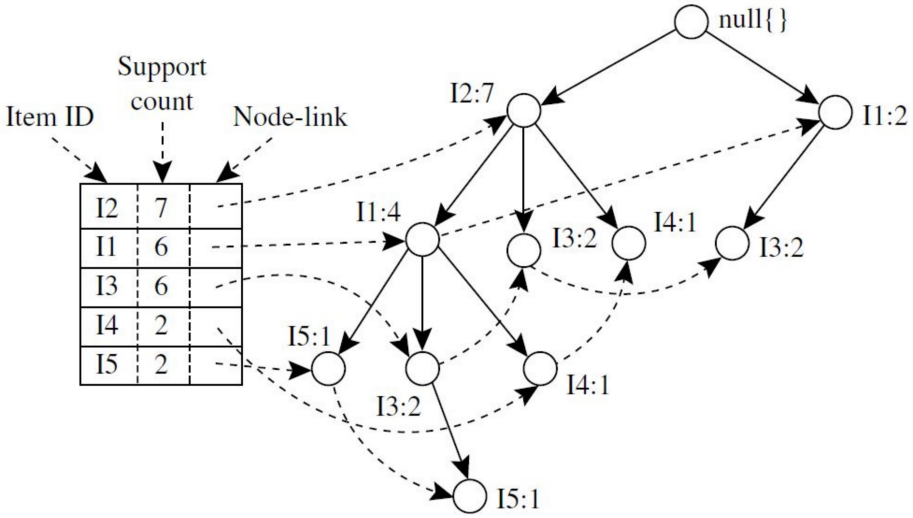
1. Transforms the database into a compressed data structure FP-tree, which retains frequent itemsets information.
2. Mines the FP-tree for frequent itemsets by:
    a. Divide it into a set of conditional databases (called conditional pattern base), each associated with one frequent item.
    b. Mines each of these patterns separately to generate frequent itemsets, without the need to re-count the original database.

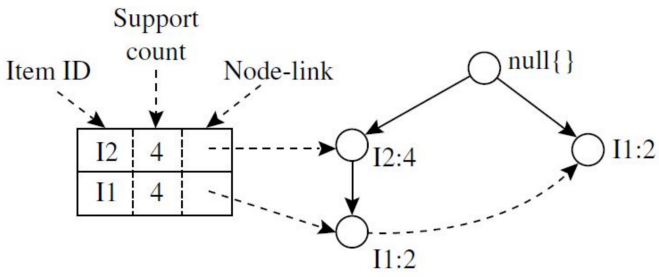
FP-growth (finding frequent itemsets without candidate generation)

Transactional Data for an *AllElectronics* Branch

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

FP-growth





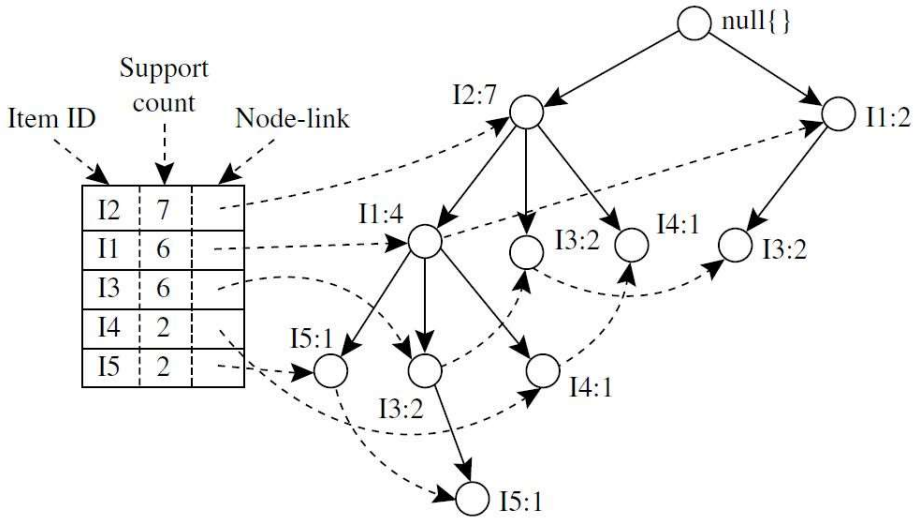
Item	*Conditional Pattern Base*	*Conditional FP-tree*	*Frequent Patterns Generated*
I5	{{I2, I1: 1}, {I2, I1, I3: 1}}	⟨I2: 2, I1: 2⟩	{I2, I5: 2}, {I1, I5: 2}, {I2, I1, I5: 2}
I4	{{I2, I1: 1}, {I2: 1}}	⟨I2: 2⟩	{I2, I4: 2}
I3	{{I2, I1: 2}, {I2: 2}, {I1: 2}}	⟨I2: 4, I1: 2⟩, ⟨I1: 2⟩	{I2, I3: 4}, {I1, I3: 4}, {I2, I1, I3: 2}
I1	{{I2: 4}}	⟨I2: 4⟩	{I2, I1: 4}

How is Conditional FP tree for I3 calculated? In video says take common

Figure 6.7, 6.8 in Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.

Transactional Data for an *AllElectronics* Branch

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3



FP-tree:

- A trie (a prefix tree) data structure.
- A compressed representation of a conditional DB.
- The path from the root to a leaf represents a repeated sub-transaction (frequent pattern) in the database.
- The path from the root to an internal node represent either a frequent pattern or a prefix.
- Each node is associated with a count representing the number of transactions in the original database that contain its path.
- The prefixes are sorted in the order from the most frequent to the least frequent to maximize the advantages of prefix-based compression.

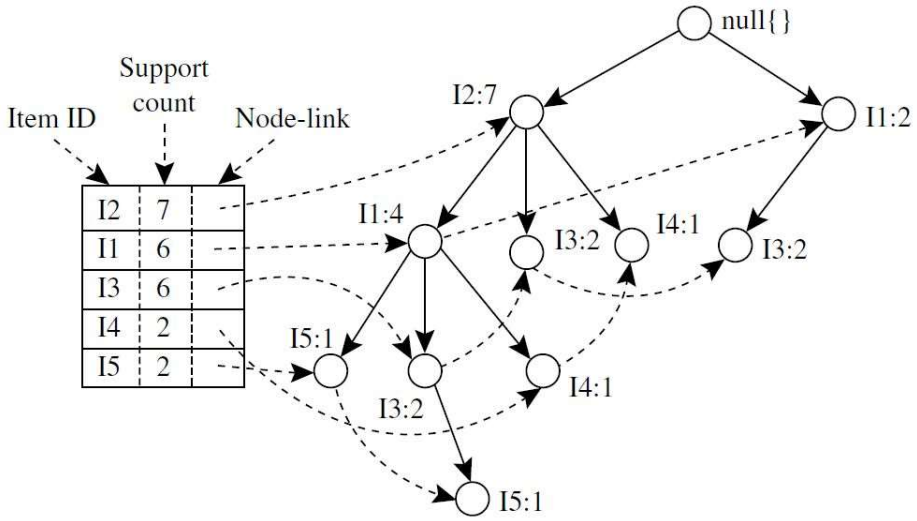
FP-Growth pseudo code

Algorithm *FP-growth*(FP-Tree of frequent items: $\mathcal{FPT}$, Minimum Support: *minsup*, Current Suffix: P)

begin
  if $\mathcal{FPT}$ is a single path
    then determine all combinations C of nodes on the path, and report $C \cup P$ as frequent;
  else (Case when $\mathcal{FPT}$ is not a single path)
  for each item i in $\mathcal{FPT}$ **do begin**
    report itemset $P_i = \{i\} \cup P$ as frequent;
    Use pointers to extract conditional prefix paths from $\mathcal{FPT}$ containing item i;
    Readjust counts of prefix paths and remove i;
    Remove infrequent items from prefix paths and reconstruct conditional FP-Tree $\mathcal{FPT}_i$;
    if $(\mathcal{FPT}_i \neq \phi)$ **then** *FP-growth*$(\mathcal{FPT}_i, minsup, P_i)$;
  end
end

Transactional Data for an *AllElectronics* Branch

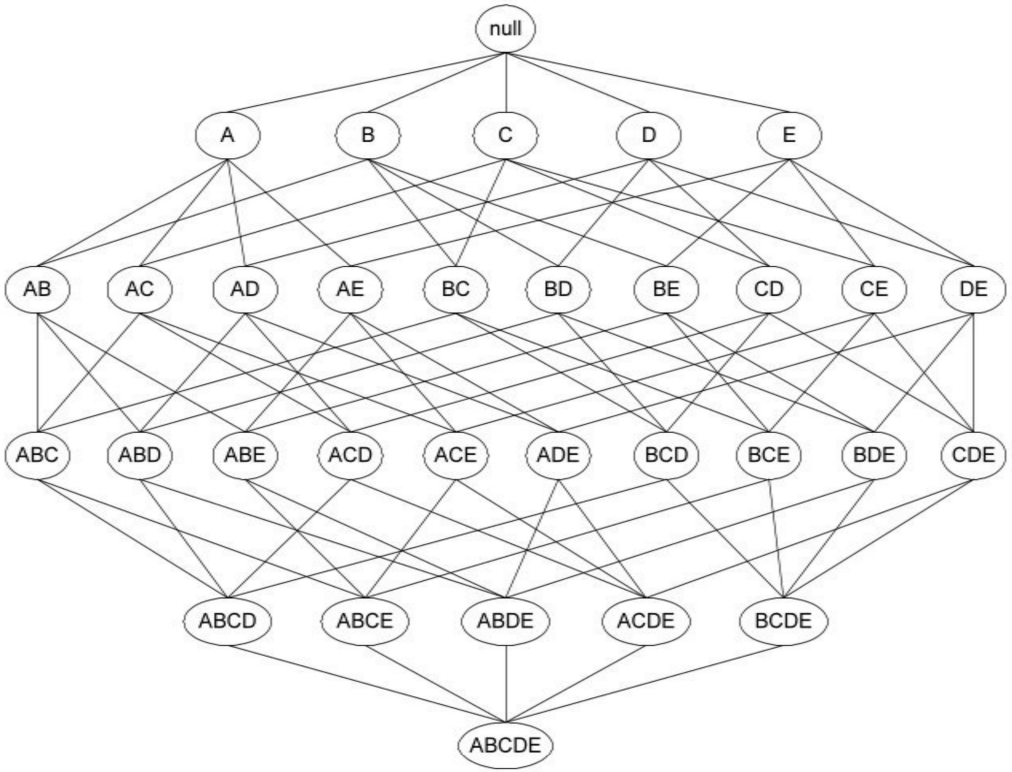
TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3



FP-Growth:

- FP-Growth is a recursive Algorithm.
- FP-Growth find all frequent itemsets ending with a particular suffix by splitting the problem into smaller sub-problems.
- Suppose we are interested in finding all frequent itemsets ending in ‘I3’.
  - First make sure that I3 is itself frequent.
  - Solve the sub-problems of finding frequent itemsets ending with I1 I3, I2 I3, I4 I3, I5 I3.
  - In turn, each of these sub-problems is recursively decomposed into smaller subproblems, until they all stop recursing.
  - Merge the solutions of these sub-problems.

Does FP-Growth traverse the lattice ? How ?



Why does FP-Growth outperform Apriori ?

FP-Growth counts the database only once
Apriori creates more candidates to be tested than FP-Growth
Apriori creates candidates, and candidate generation can be expensive
Space complexity of FP-Growth is smaller, because no candidate generation/storage is needed
FP-Growth compresses the database, thus dealing with a smaller structure, on which we can mine the pattern directly

Evaluating Association Rules

- Support-confidence
- Lift
- Correlation analysis.
- IS Measure.
- etc

Table 5.9. Examples of objective measures for the itemset $\{A, B\}$.

Measure (Symbol)	Definition
Correlation (ϕ)	$\frac{Nf_{11}-f_{1+}f_{+1}}{\sqrt{f_{1+}f_{+1}f_{0+}f_{+0}}}$
Odds ratio (α)	$(f_{11}f_{00})/(f_{10}f_{01})$
Kappa (κ)	$\frac{Nf_{11}+Nf_{00}-f_{1+}f_{+1}-f_{0+}f_{+0}}{N^2-f_{1+}f_{+1}-f_{0+}f_{+0}}$
Interest (I)	$(Nf_{11})/(f_{1+}f_{+1})$
Cosine (IS)	$(f_{11})/(\sqrt{f_{1+}f_{+1}})$
Piatetsky-Shapiro (PS)	$\frac{f_{11}}{N} - \frac{f_{1+}f_{+1}}{N^2}$
Collective strength (S)	$\frac{f_{11}+f_{00}}{f_{1+}f_{+1}+f_{0+}f_{+0}} \times \frac{N-f_{1+}f_{+1}-f_{0+}f_{+0}}{N-f_{11}-f_{00}}$
Jaccard (ζ)	$f_{11}/(f_{1+}+f_{+1}-f_{11})$
All-confidence (h)	$\min\left[\frac{f_{11}}{f_{1+}}, \frac{f_{11}}{f_{+1}}\right]$

Michael Steinbach, Pang-Ning Tan, and Vipin Kumar, Introduction to Data Mining, Pearson 2005.
https://www-users.cs.umn.edu/~kumar001/dmbook/ch5_association_analysis.pdf

Limitation of Support-confidence

What is the limitation?

Consider this tea-coffee example, How strong is the rule:

{Tea} => {Coffee} ?

Conf({Tea} => {Coffee})= 150/200= 75% (high confidence)

But Sup(coffee)=80%

So actually drinking tea decreases the probability of drinking coffee !!!

How strong is the rule:

{Tea} => {Honey} ?

	B	¬B	Total
A	f11	f10	f1-
¬A	f01	f00	f0-
Total	f-1	f-0	

	Coffee	¬Coffee	Total
Tea	150	50	200
¬Tea	650	150	800
Total	800	200	

	Honey	¬Honey	Total
Tea	100	100	200
¬Tea	20	780	800
Total	120	880	

Lift Sup(A U B)/ (Sup(A) * Sup(B))

How strong is the rule:

{Tea} => {Coffee} ?

Lift({Tea} => {Coffee})=

0.15/(0.2*0.8) = 0.15/0.16 < 1

Tea and Coffee negatively correlated

How strong is the rule:

{Tea} => {Honey} ?

	B	¬B	Total
A	f11	f10	f1-
¬A	f01	f00	f0-
Total	f-1	f-0	

	Coffee	¬Coffee	Total
Tea	150	50	200
¬Tea	650	150	800
Total	800	200	

	Honey	¬Honey	Total
Tea	100	100	200
¬Tea	20	780	800
Total	120	880	

Limitation of Lift

Using Lift, How strong is the rule:

{p} => {q} ?

Lift({p} => {q})= 0.88/(0.93*0.93) = 1.02

How strong is the rule:

{r} => {s} ?

Lift({r} => {s})= 0.02/(0.07*0.07) = 4.08

But p,q appear together 88% of the time, while r,s appear together only 2% of the time.

Confidence is a better indicator than Lift in this case.

	p	¬p	Total
q	880	50	930
¬q	50	20	70
Total	930	70	

	r	¬r	Total
s	20	50	70
¬s	50	880	930
Total	70	930	

Correlation Analysis

- $\Phi = \frac{(f11 * f00 - f10 * f01)}{\sqrt{f\_1 * f\_0 * f1\_ * f0\_}}$
- -1 perfect negative correlation,
  1 perfect positive correlation,
  0 statistically independent.

- How strong is the rule:

  {p} => {q} ?

= (0.88*0.02) –(0.05*0.05)
/(0.93*0.93*0.07*0.07) = 0.232

	p	¬p	Total
q	880	50	930
¬q	50	20	70
Total	930	70	

	r	¬r	Total
s	20	50	70
¬s	50	880	930
Total	70	930	

How to choose a measure ?

- There are many more measures beyond what we have studied.
- These measures are not consistent. Applying two measures to the same set of rules, yield different sorting of rules from most interesting to least interesting.
- A good choice must be based on a clear understanding of the measure and its properties (inversion invariance, null addition invariance, etc).
- All these measures all called objective measures. They are good for automatic filtering of rules.
- One might additionally need to assess the rules using interactive visualizations, subjective measures based on domain experience, and template based search (i.e., report only the rules that can increase the sales of bio products).

Multi-level Association Rule Mining

Table 7.1 Task-Relevant Data, *D*

TID	*Items Purchased*
T100	Apple 17″ MacBook Pro Notebook, HP Photosmart Pro b9180
T200	Microsoft Office Professional 2010, Microsoft Wireless Optical Mouse 5000
T300	Logitech VX Nano Cordless Laser Mouse, Fellowes GEL Wrist Rest
T400	Dell Studio XPS 16 Notebook, Canon PowerShot SD1400
T500	Lenovo ThinkPad X200 Tablet PC, Symantec Norton Antivirus 2010
…	…

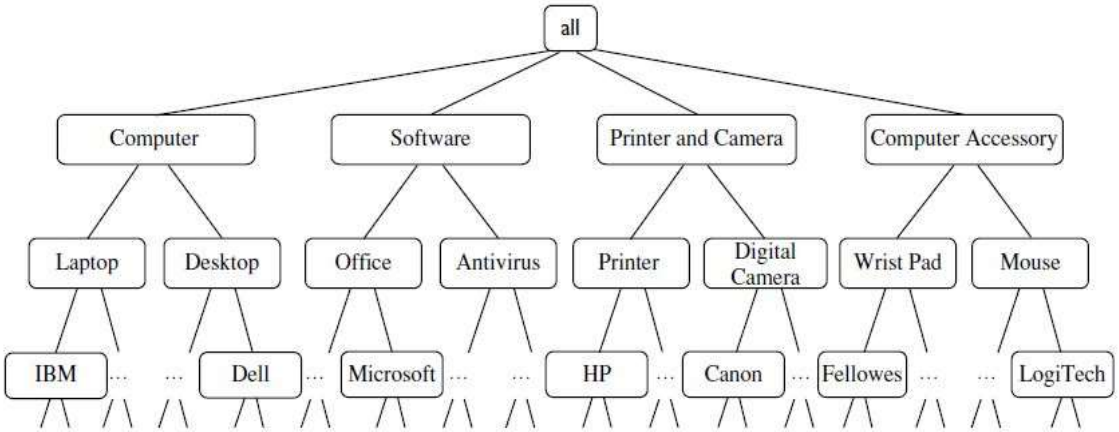


Figure 7.2 Concept hierarchy for *AllElectronics* computer items.

Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.

Concept Hierarchies

- A concept hierarchy defines a sequence of mappings from a set of low-level concepts to a higher-level, more general concept set.
- Data can be generalized by replacing low-level concepts within the data by their corresponding higher-level concepts, or *ancestors*, from a concept hierarchy.
- level 0 at the root node is the most general abstraction level). Level k at the leaves is the most specific abstraction level.
- Concept hierarchies for numeric attributes can be generated using discretization techniques (data preprocessing).
- Concept hierarchies may be specified by users familiar with the data such as store managers in the case of the example.

Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.

Multi-level Association Rule Mining

- Rules that are generated at multiple abstraction levels.
- In general, a top-down strategy is employed, where counts are accumulated for the calculation of frequent itemsets at each concept level, starting at concept level 1 and working downward in the hierarchy toward the more specific concept levels, until no more frequent itemsets can be found.
- For each level, any algorithm for discovering frequent itemsets may be used, such as Apriori or its variations.

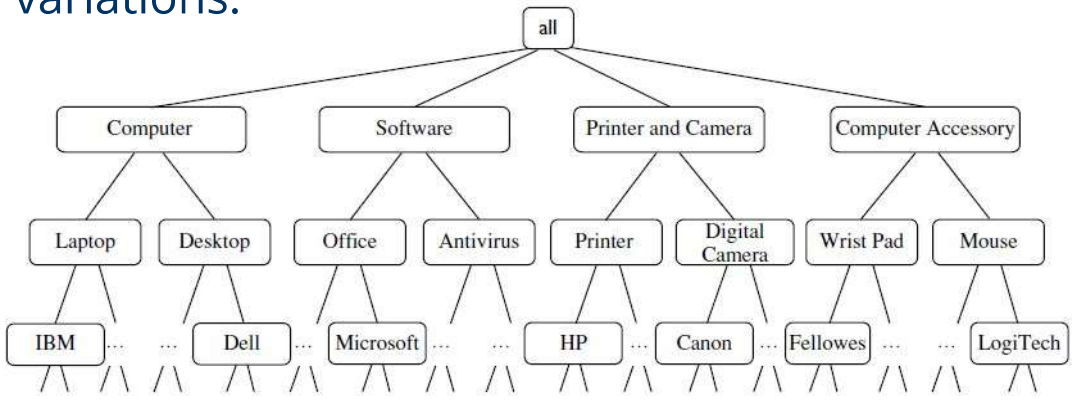


Figure 7.2 Concept hierarchy for *AllElectronics* computer items.

Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.

Multi-level Association Rule Mining

- How to set the support threshold ?
- Using same support for all levels
  - + Simple for users as it requires a single parameter (minsup).
  - + Apriori (or alternative) can do pruning using the parent-child relationship. If a parent is not frequent, … ?
  - If minsup is too high, only higher levels of abstraction can generate rules.
  - If minsup is too low, redundant and uninteresting rules are generated.

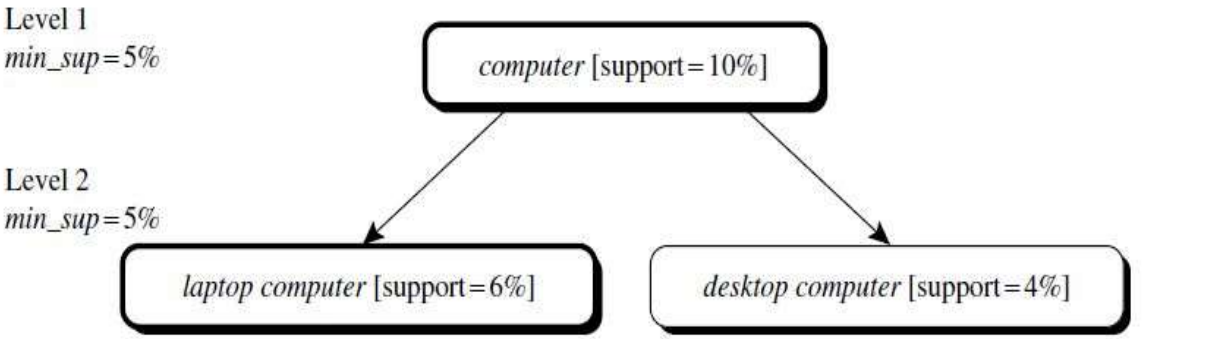


Figure 7.3 Multilevel mining with uniform support.

Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.

Multi-level Association Rule Mining

- How to set the support threshold ?
  - Using reduced minsup at lower levels
  - Using group-based support
    - Domain experts know the frequently sold items and the rare ones.
    - Set higher minsup for the frequently sold items and lower minsup for the rarely sold ones.
    - For example: Cameras > 1000$ are rarely sold. Interesting rules that involve this item can be lost if a minsup that considers the average is used.
    - But Apriori accepts a single minsup parameter ???

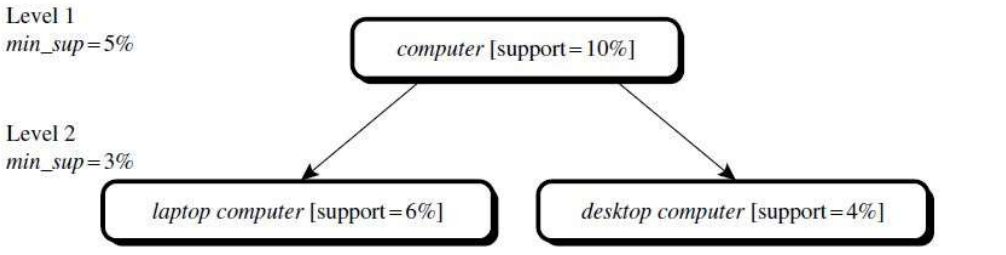


Figure 7.4 Multilevel mining with reduced support.

Multi-Dimenstional Association Rule Mining

- So far we have studied rules in the form:
  - buys(X, Camera) => buys(X, Printer).
- These rules are mined over the dimension “buys”.
- What about:
  - age(X, “20-29”) and occupation(X, “student”) => buys(X, “laptop”).
- Basically the mining is done in the same way, except that the counting is changed from counting occurrences of items into counting predicate fulfillment.
- This is typical for mining association rules from transactional databases.

Frequent Pattern Mining Applications

- Market basket analysis.
- Noise filtering (data cleaning):
  - Frequent itemsets are less probable to be noise.
- Data reduction (compression):
  - Very frequent itemsets are more probable to be non-interesting (e.g., stop words), and can be removed.
- Recommender systems:
  - Based on doing X, the system recommends doing Y.
- Cluster discovery:
  - Co-authors in DBLP.

Credits and Readings

These slides, except when explicitly stated, use material from:

- Charu C.Aggarwal. Data Mining The Textbook, Springer.
- Jiawei Han, Micheline Kamber and Jian Pe, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2012.
- Michael Steinbach, Pang-Ning Tan, and Vipin Kumar, Introduction to Data Mining, Pearson 2005.