Association Analysis

Unit 3

Index

Frequent Itemset Mining

Apriori Algorithm

Rule generation

FP-Growth Algorithm

Compact Representation of Frequent Item sets

Maximal Frequent Item sets

Closed Frequent Item sets

Introduction

Association analysis, often applied to market basket data,

Is a methodology used to extract valuable insights and relationships hidden within large datasets.

The process involves

identifying frequent itemset

association rules

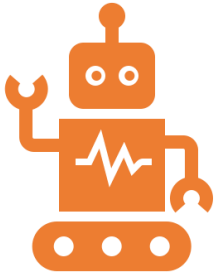
To understand customer behaviour, support business applications like

Marketing strategies

Inventory management

Customer relationship management.

Application Beyond Retail



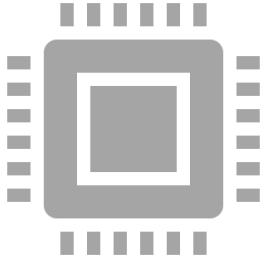
Retail Use Cases:

Assist in marketing strategies,

Advertisement placement

Store layout optimization

Inventory management.



Diverse Applications:

Used in telecommunications for fault prediction and various other data analysis tasks.

The Market Basket Analysis

- A grocery store chain keeps a record of weekly transactions where each transaction represents the items bought during one cash register transaction.
- The executives of the chain receive a summarized report of the transactions indicating what types of items have sold at what quantity.
- In addition, they periodically request information about what items are commonly purchased together.
- They find that 100% of the time that Peanut Butter is purchased, so is Bread.
- In addition, 33.3% of the time Peanut Butter is purchased, Jelly is also purchased.
- However, Peanut Butter exists in only about 50% of the overall transactions.

Market basket Transaction

Table 4.1. An example of market basket transactions.

TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}

Preliminaries

1. **Binary Representation of Market Basket Data**
2. **Itemset and Support Count**
3. **Frequent Itemsets and Association Rules**
4. **Importance of Support and Confidence**
5. **Causality vs. Association**

Binary Representation of Market Basket Data

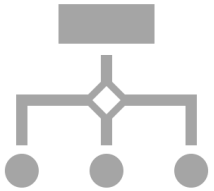
- **Representation**:
  - Each row represents a transaction
  - Each column corresponds to an item.
- **Binary Format**:
  - Items are treated as binary variables,
  - taking the value 1 if present in a transaction and 0 if absent.
- **Asymmetric Binary Variables**:
  - Items are treated as asymmetric binary variables because their presence is considered more crucial than absence in transactions.

Table 4.2. A binary 0/1 representation of market basket data.

TID	Bread	Milk	Diapers	Beer	Eggs	Cola
1	1	1	0	0	0	0
2	1	0	1	1	1	0
3	0	1	1	1	0	1
4	1	1	1	1	0	0
5	1	1	1	0	0	1

Itemset and Support Count

An itemset is a collection of zero or more items from a dataset.



The support count of an itemset is the number of transactions in which that itemset occurs.

Support is calculated as the fraction of transactions in which an itemset occurs.

Itemset and Support Count

Let $I = \{i_1, i_2, \ldots, i_d\}$ be the set of all items in a market basket data and $T = \{t_1, t_2, \ldots, t_N\}$ be the set of all transactions.

Each transaction t_i contains a subset of items chosen from I. In association analysis, a collection of zero or more items is termed an itemset.

If an itemset contains k items, it is called a k-itemset.

For instance, {Beer, Diapers, Milk} is an example of a 3-itemset. The null (or empty) set is an itemset that does not contain any items.

A transaction t_j is said to contain an itemset X if X is a subset of t_j .

Itemset and Support Count

For example, the second transaction shown in Table 4.2 contains the itemset {Bread, Diapers} but not {Bread, Milk}.

An important property of an itemset is its support count, which refers to the number of transactions that contain a particular itemset.

Mathematically, the support count, $\sigma(X)$, for an itemset X can be stated as follows:

$$\sigma(X) = \left| \{ t_i \mid X \subseteq t_i, t_i \in T \} \right|$$

where the symbol $|\ .\ |$ denotes the number of elements in a set.

Frequent Item sets

An itemset is considered frequent if its support exceeds a user-defined threshold, called minsup.

Frequent item sets are of interest because they represent patterns that occur frequently in the dataset.

In the data set shown in Table 4.2, the support count for {Beer, Diapers, Milk} is equal to two because there are only two transactions that contain all three items.

Often, the property of interest is the support, which is fraction of transactions in which an itemset occurs:

$$s(X) = \sigma(X)/N$$

An itemset X is called frequent if s(X) is greater than some user-defined threshold, minsup.

Association Rules

01

An association rule is an implication of the form X → Y, where X and Y are disjoint itemsets.

02

Support and confidence are metrics used to measure the strength of association rules.

03

Support for a rule X → Y measures how often X and Y appear together in the dataset.

04

Confidence for a rule X → Y measures the reliability of the inference made by the rule.

The formal definitions of these metrics

$$\text{Support, } s(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{N};$$

$$\text{Confidence, } c(X \longrightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}.$$

Example

- Consider the rule {Milk, Diapers} −→ {Beer}.
- the support count for {Milk, Diapers, Beer} is 2
- the total number of transactions is 5.

rule's support = 2/5 = 0.4

- The rule's confidence is obtained by dividing the support count for {Milk, Diapers, Beer} by the support count for {Milk, Diapers}.
- Since there are 3 transactions that contain milk and diapers, the

confidence for this rule = 2/3 = 0.67

Importance of Support and Confidence

Support:

Identifies rules frequently applicable in a dataset, helping distinguish rules with significant occurrences from chance findings.

Confidence:

Measures the reliability of an inference made by a rule.

Higher confidence indicates a higher likelihood of Y in transactions containing X.

Caution in Interpretation:

Association rules indicate co-occurrence, not causation.

Understanding causality requires knowledge about cause-and-effect relationships.

Significance

Thresholds:

User-defined thresholds (minsup) aid in identifying meaningful and statistically significant rules.

Business Relevance:

Rules with low support might not be profitable for businesses, hence the focus on rules with higher support.

Interpretation and Causality:

Association rules don't imply causality; they reflect co-occurrence relationships.

QUESTION

1. For each of the following questions, provide an example of an association rule from the market basket domain that satisfies the following conditions. Also, describe whether such rules are subjectively interesting.

(a) A rule that has high support and high confidence.

(b) A rule that has reasonably high support but low confidence.

(c) A rule that has low support and low confidence.

(d) A rule that has low support and high confidence.

Customer ID	Transaction ID	Items Bought
1	0001	$\{a, d, e\}$
1	0024	$\{a, b, c, e\}$
2	0012	$\{a, b, d, e\}$
2	0031	$\{a, c, d, e\}$
3	0015	$\{b, c, e\}$
3	0022	$\{b, d, e\}$
4	0029	$\{c, d\}$
4	0040	$\{a, b, c\}$
5	0033	$\{a, d, e\}$
5	0038	$\{a, b, e\}$

QUESTION

- Consider the data set shown

a) Compute the support for itemsets {e}, {b, d}, and {b, d, e} by treating each transaction ID as a market basket.

b) Use the results in part (a) to compute the confidence for the association rules {b, d} −→ {e} and {e} −→ {b, d}. Is confidence a symmetric measure?

c) Repeat part (a) by treating each customer ID as a market basket. Each item should be treated as a binary variable (1 if an item appears in at least one transaction bought by the customer, and 0 otherwise.)

d) Use the results in part (c) to compute the confidence for the association rules {b, d} −→ {e} and {e} −→ {b, d}.

Formulation of the Association Rule Mining Problem

The association rule mining problem aims to find rules that have certain minimum thresholds for support and confidence within a dataset of transactions.

Support represents how frequently an itemset or rule appears in the dataset, while confidence measures the reliability of the rule's implication.

The definition (Association Rule Discovery) formalizes this problem:

given a set of transactions T, the objective is to identify all rules that meet the specified support (minsup) and confidence (minconf) thresholds.

Brute-Force Approach:

- The direct approach of computing support and confidence for every possible rule is computationally impractical due to the exponential number of rules that can be generated from a dataset.
- The equation represents the total number of potential rules that can be derived from a dataset with d items:

$$R = 3^d - 2^{d+1} + 1$$

- Even with relatively small datasets, this approach involves a huge number of computations.

Decoupling Support and Confidence

An important observation is that the support of a rule is equivalent to the support of its associated itemset.

This observation suggests a strategy:

if an itemset is infrequent (below the minsup threshold), all candidate rules involving that itemset can be discarded without calculating their confidence values.

This observation helps to prune potential rules early, reducing computational efforts.

Decomposition of Association Rule Mining

To mitigate the computational complexity, association rule mining is divided into two primary tasks:

- Identifying all item sets that meet the minimum support threshold.

- Extracting high-confidence rules from the frequent itemsets obtained in the previous step.
- These rules, termed strong rules, meet both the support and confidence criteria.

Frequent Itemset Generation

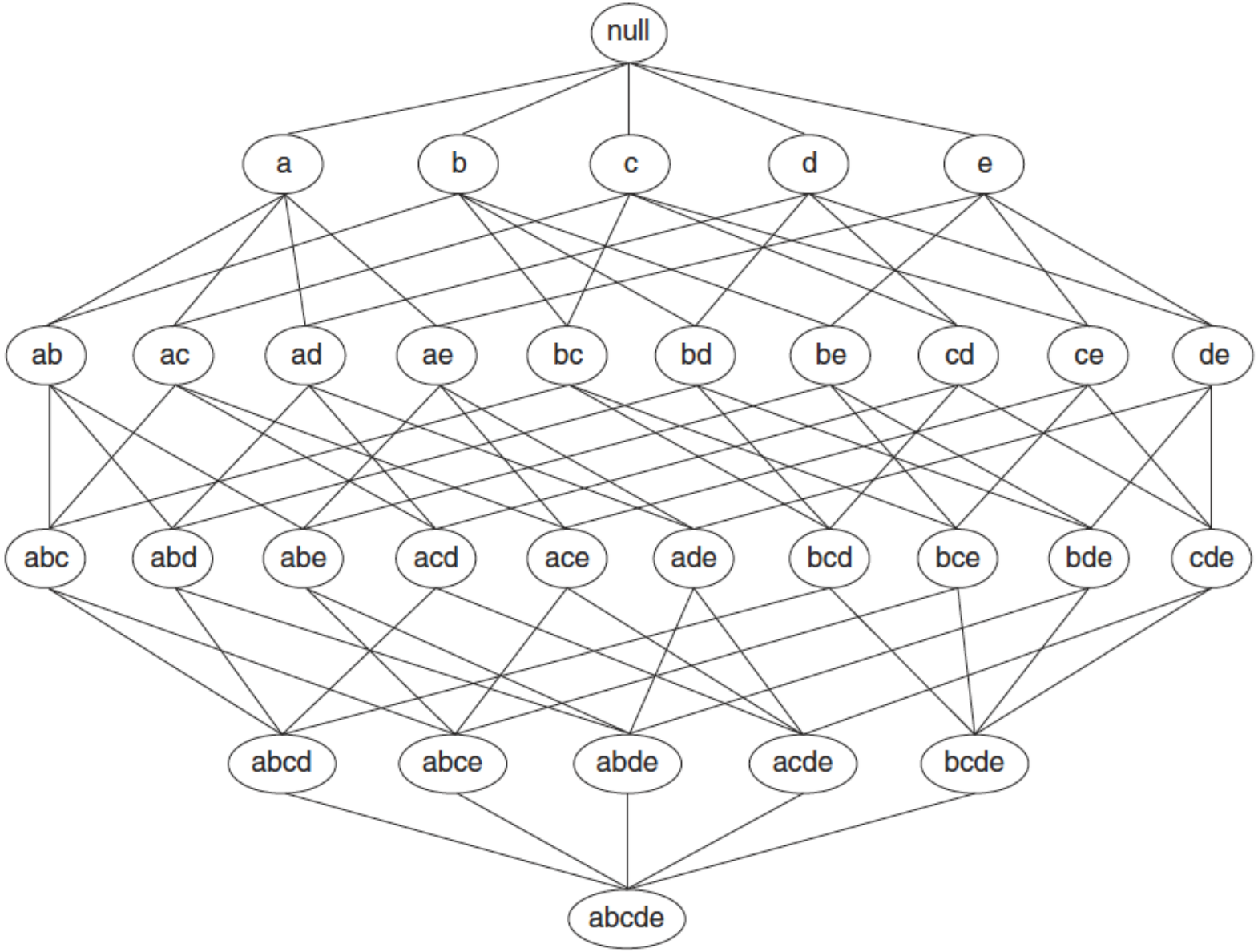


Figure 4.1. An itemset lattice.

Frequent Itemset Generation challenges

Lattice Structure and Itemset Enumeration:

- The lattice structure in association rule mining represents a systematic way to organize and visualize relationships between different itemsets based on a given dataset.
- Figure 4.1 illustrates an example of an itemset lattice for a set of **items I = {a, b, c, d, e}** It demonstrates the hierarchical arrangement of itemsets based on inclusion relationships.

Exponential Growth in Itemsets:

- The number of potential itemsets grows exponentially with the number of items in the dataset.
- For a dataset with k items, the total number of possible itemsets, excluding the null set, can be $2^k - 1$
- This exponential growth significantly increases the search space of potential itemsets.

Frequent Itemset Generation challenges

Brute-force methods involve calculating the support count for each candidate itemset in the lattice structure.

To compute the support count, each candidate itemset needs to be checked against every transaction in the dataset.

For a dataset with N transactions and M candidate itemsets ($M=2^K-1$), this approach requires O(NMw) comparisons:

- N is the number of transactions.
- M is the number of candidate itemsets.
- w is the maximum transaction width (maximum number of items in a transaction).

The comparison involves checking if a candidate itemset is present in each transaction and incrementing its support count accordingly.

Frequent Itemset Generation challenges

Computational Complexity and Scalability:

- The brute-force approach becomes computationally expensive, especially for larger datasets or a higher number of items (larger values of k).
- As the number of candidate itemsets and transactions increases, the number of comparisons grows significantly, leading to increased computational requirements.

Need for Efficient Algorithms:

- Efficient algorithms like Apriori, FP-Growth, or Eclat have been developed to address the challenges posed by the exponential growth in itemsets.
- These algorithms aim to reduce the search space, optimize the process of finding frequent itemsets, and make association rule mining computationally feasible for practical applications.

Reducing the computational complexity of frequent itemset generation

Reduce the Number of Candidate Itemsets (M) using the Apriori Principle

Reduce the Number of Comparison

Reduce the Number of Transactions (N)

The Apriori Principle

https://youtu.be/WGlMlS_Yydk?si=mLqVw4y_PNt9dLip

Apriori Algorithm (Associated Learning) - Fun and Easy Machine Learning

205K views • 6 years ago

Augmented Startups

Do you know what **Apriori Algorithms** are and how to use it for **machine learning**? Watch this video to find out. ⭐ Buy Me Coffee ...

Introduction | Coffee Dataset | What is Apriori | Example | Support | Conference | Intuitive... 10 chapters

The Apriori Principle

- If an itemset is frequent, then all of its subsets must also be frequent.
- Consider the itemset lattice shown in Figure 4.3. Suppose {c, d, e} is a frequent itemset.
- Clearly, any transaction that contains {c, d, e} must also contain its subsets
  - {c, d}, {c, e}, {d, e}, {c}, {d}, and {e}
- As a result, if {c, d, e} is frequent, then all subsets of {c, d, e} (i.e., the shaded item sets in this figure) must also be frequent.

Apriori

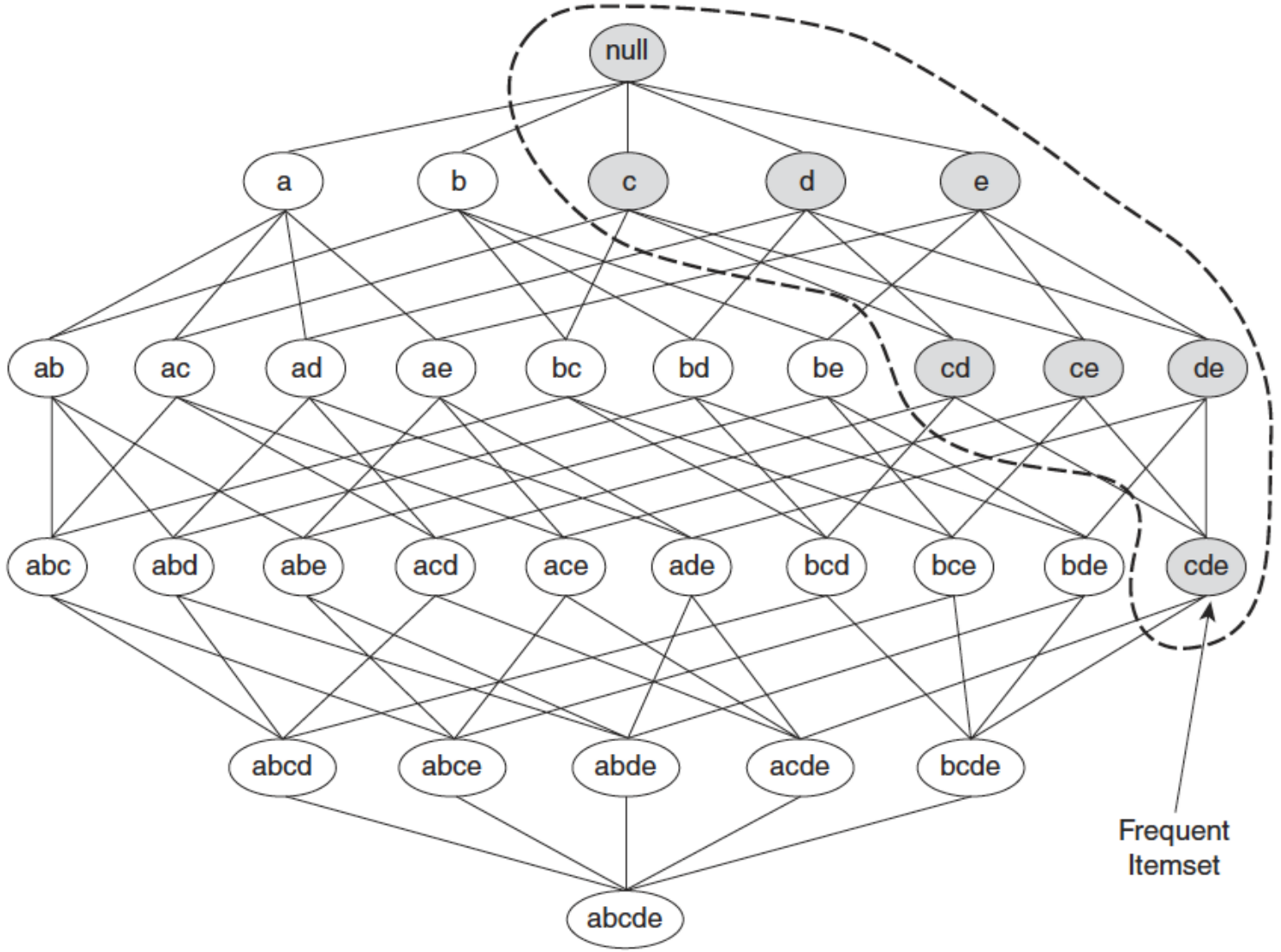


Figure 4.3. An illustration of the *Apriori* principle. If $\{c, d, e\}$ is frequent, then all subsets of this itemset are frequent.

Anti-monotone Property of Support Measure

The support measure possesses an anti-monotone property

- Which means that the support for an itemset never exceeds the support for its subsets.

This property is expressed as:

- A measure f possesses the antimonotone property if for every itemset X that is a proper subset of itemset Y , i.e. $X \subset Y$, we have $f(Y) \leq f(X)$.

Support based Pruning

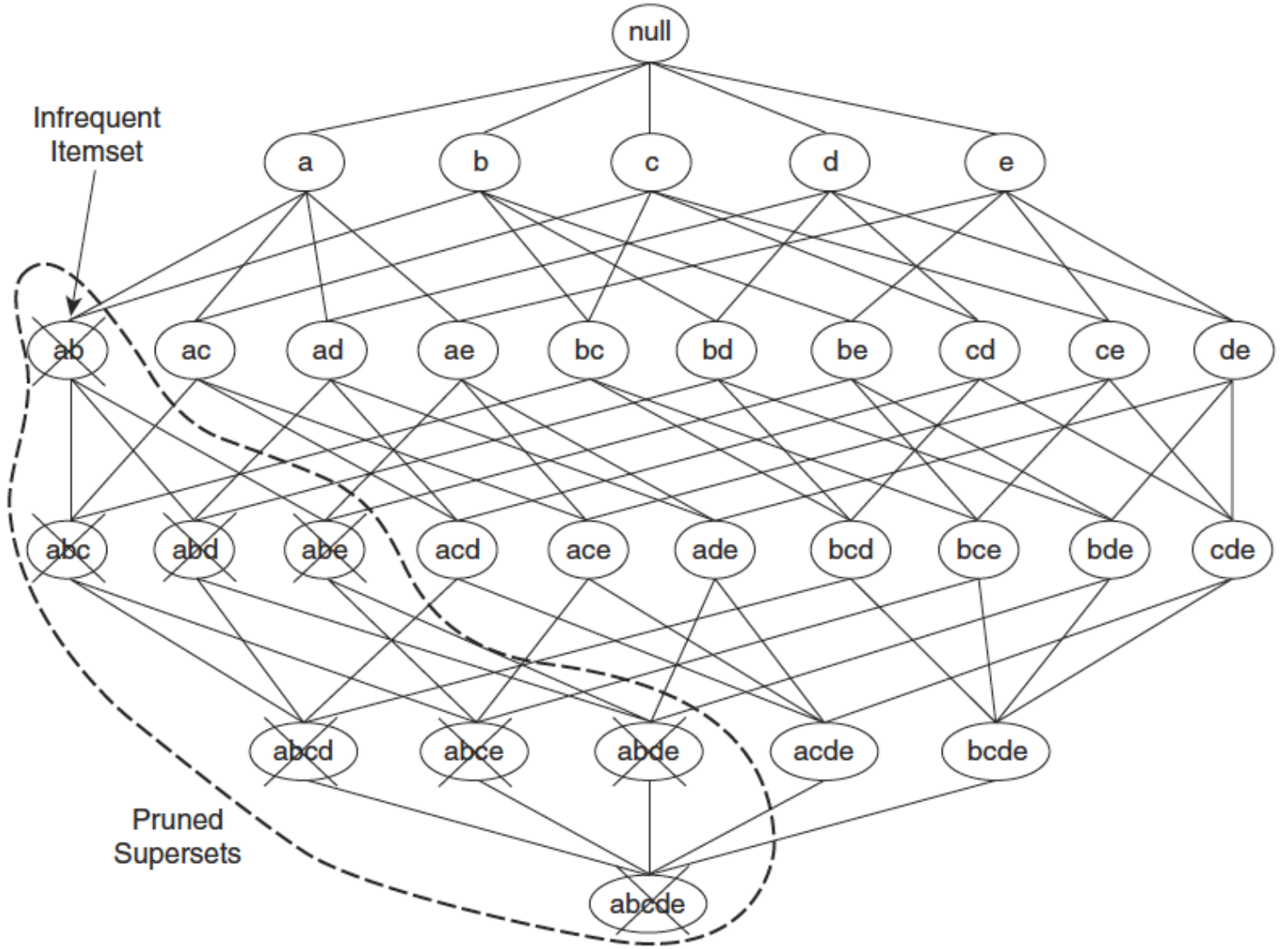


Figure 4.4. An illustration of support-based pruning. If $\{a, b\}$ is infrequent, then all supersets of $\{a, b\}$ are infrequent.

Anti-monotone Property of Support Measure

If {a, b} is determined to be infrequent, then any set containing {a, b} as a subset (superset of {a, b}) will also be infrequent.

This property enables support-based pruning in algorithms like Apriori, where the search space is significantly reduced by eliminating the need to explore supersets of infrequent item sets.

Support-based pruning helps efficiently manage the exponential growth of potential itemsets by reducing unnecessary computations.

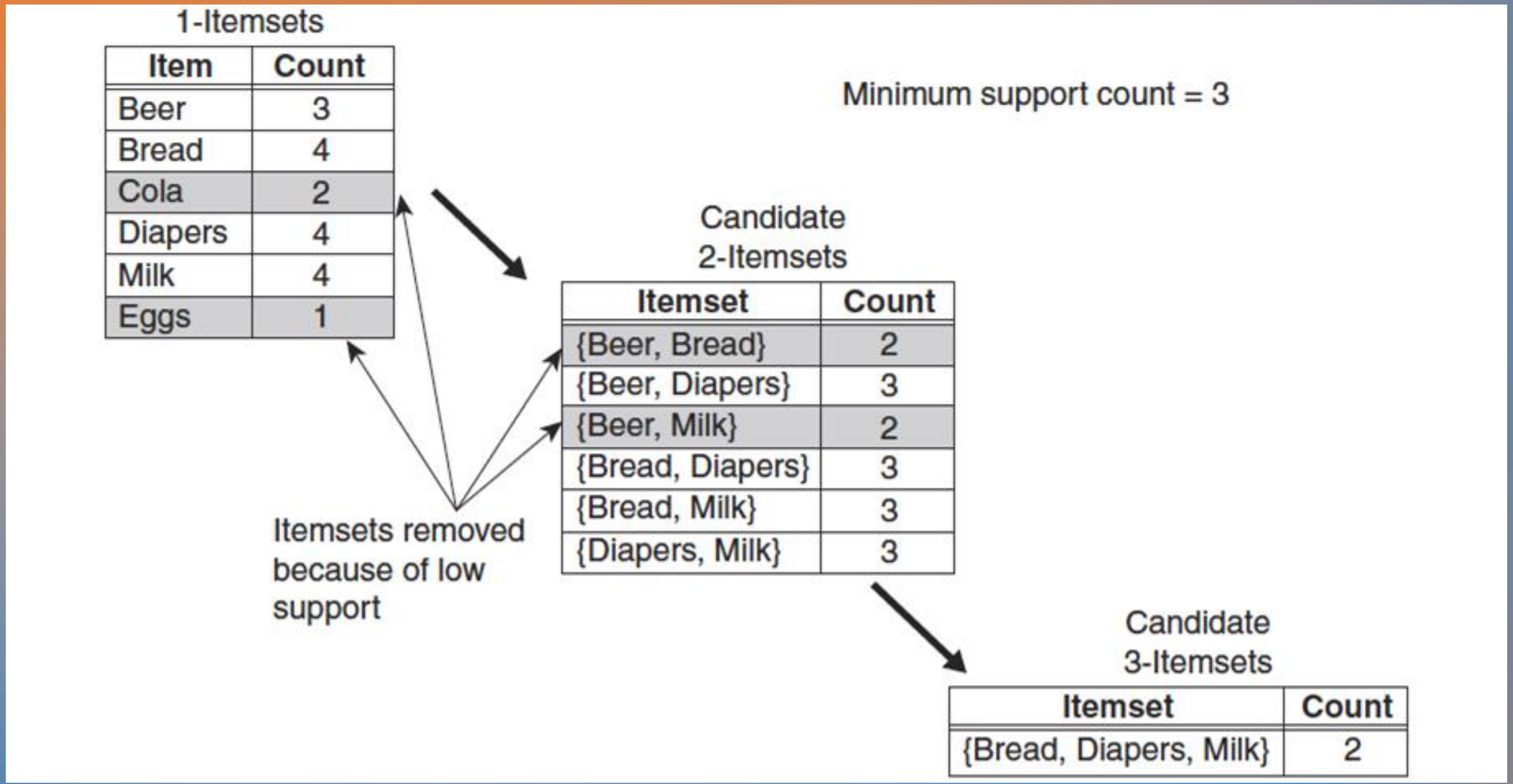
Once an itemset fails to meet the minimum support threshold, there's no need to consider its supersets, as they are guaranteed to have even lower support.

Frequent Itemset Generation in the Apriori Algorithm

Apriori: First association rule mining algorithm employing support-based pruning.

Controls exponential growth of candidate itemsets to find frequent itemsets.

Illustration in Figure 4.5 uses market basket transactions from Table 4.1.


1-Itemsets
Item
Count
Beer
3
Bread
4
Cola
2
Diapers
4
Milk
4
Eggs
1
Minimum support count = 3
Candidate
2-Itemsets
Itemset
Count
{Beer, Bread}
2
{Beer, Diapers}
3
{Beer, Milk}
2
{Bread, Diapers}
3
{Bread, Milk}
3
{Diapers, Milk}
3
Itemsets removed
because of low
support
Candidate
3-Itemsets
Itemset
Count
{Bread, Diapers, Milk}
2

Basic Workflow

Support Threshold:

- Set at 60%, equivalent to a minimum support count of 3.

Candidate Generation:

- Initially, every item is a candidate 1-itemset.
- Discard itemsets {Cola} and {Eggs} with less than three appearances.
- Generate candidate 2-itemsets from frequent 1-itemsets.
- Use Apriori principle: Infrequent 1-itemsets lead to infrequent supersets.
- Compute support; some 2-itemsets are found infrequent.
- Remaining 4 frequent 2-itemsets used to generate candidate 3-itemsets.

Support-Based Pruning:

- Without pruning, 20 candidate 3-itemsets can be formed.
- Apply Apriori principle: Keep candidate 3-itemsets with all subsets frequent.
- Only {Bread, Diapers, Milk} meets this criterion but is itself not frequent.

Effectiveness of Pruning:

- Brute-force: Enumerating all itemsets (up to size 3) produces 41 candidates.
- Apriori principle reduces it to 13 candidates, a 68% reduction

Apriori Algorithm's Pseudocode

Determine	Determine support for each item (F1).
Generate and prune	Generate and prune k-itemsets iteratively based on (k−1)-itemsets.
Count	Count support for candidates.
Eliminate	Eliminate candidates with support counts less than N × minsup.
Terminate	Terminate when no new frequent itemsets are generated (Fk = Ø).

Characteristics of Apriori Algorithm

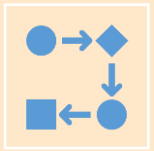
Level-Wise:

Traverses itemset lattice from frequent 1-itemsets to the maximum size.

Generate-and-Test Strategy:

Generates new candidate itemsets at each level from previous frequent itemsets.

Counts support against minsup threshold.



Total Iterations:

kmax+1, where kmax is the maximum size of frequent itemsets.

Table 6.3. Example of market basket transactions.

Transaction ID	Items Bought
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	$\{b, d\}$

QUESTIONS

- (a) Draw an itemset lattice representing the data set given in Table 6.3. Label each node in the lattice with the following letter(s):
  - N: If the itemset is not considered to be a candidate itemset by the Apriori algorithm. There are two reasons for an itemset not to be considered as a candidate itemset:
    (1) it is not generated at all during the candidate generation step, or
    (2) (2) it is generated during the candidate generation step but is subsequently removed during the candidate pruning step because one of its subsets is found to be infrequent.
  - F: If the candidate itemset is found to be frequent by the Apriori algorithm.
  - I: If the candidate itemset is found to be infrequent after support counting.
- (b) What is the percentage of frequent itemsets (with respect to all itemsets in the lattice)?

ANSWER

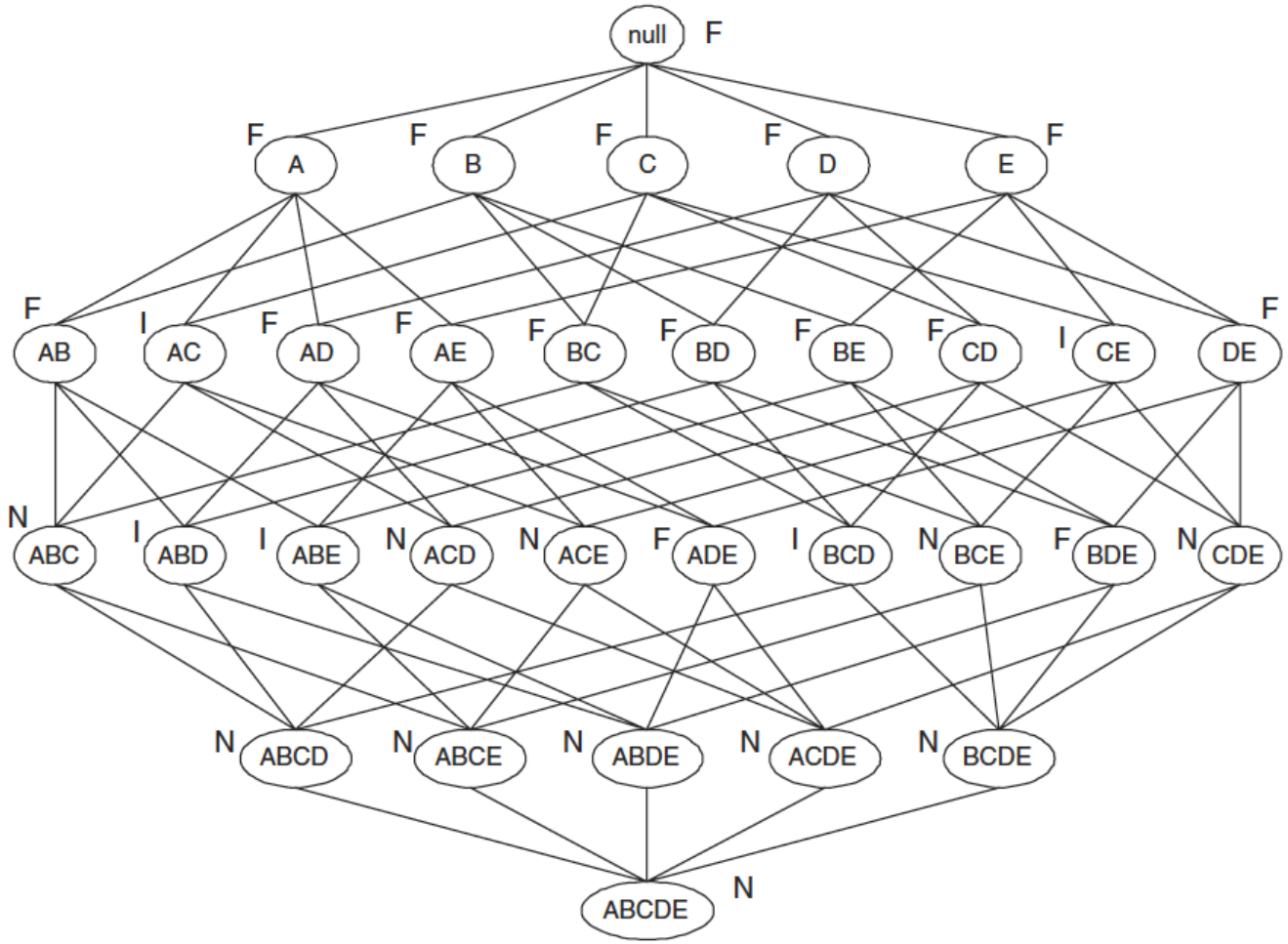


Figure 6.1. Solution.

Association Rule Generation

Extraction from Frequent Itemsets

- Frequent k-itemsets can generate up to $2^k - 2$ association rules, avoiding rules with empty antecedents or consequents.
- Rules are extracted by dividing the itemset into two non-empty subsets (X and Y-X) meeting the confidence threshold.
- All rules generated must have already met the support threshold as they are derived from frequent itemsets.

Confidence Calculation

- Confidence computation for association rules doesn't require additional scans of the transaction dataset.
- If an itemset is frequent, its subsets are also frequent due to the anti-monotone property of support. Thus, confidence calculation uses support counts obtained during frequent itemset generation.

Confidence-Based Pruning

- Confidence lacks the anti-monotone property like support. However, when comparing rules generated from the same frequent itemset, a specific confidence theorem holds.
- The theorem states that if a rule X -> Y-X doesn't meet the confidence threshold, then any subset -> Y- , where is a subset of X, won't meet the confidence threshold either.

Rule Generation in Apriori Algorithm

The Apriori algorithm adopts a level-wise approach to generate association rules, where each level corresponds to the number of items in the rule consequent.

Initially, high-confidence rules with a single item in the consequent are extracted. These rules are then used to create new candidate rules by merging consequents.

If any node in the lattice structure (representing rules generated from frequent itemsets) has low confidence, the entire subgraph can be pruned

Rule Generation in Apriori Algorithm

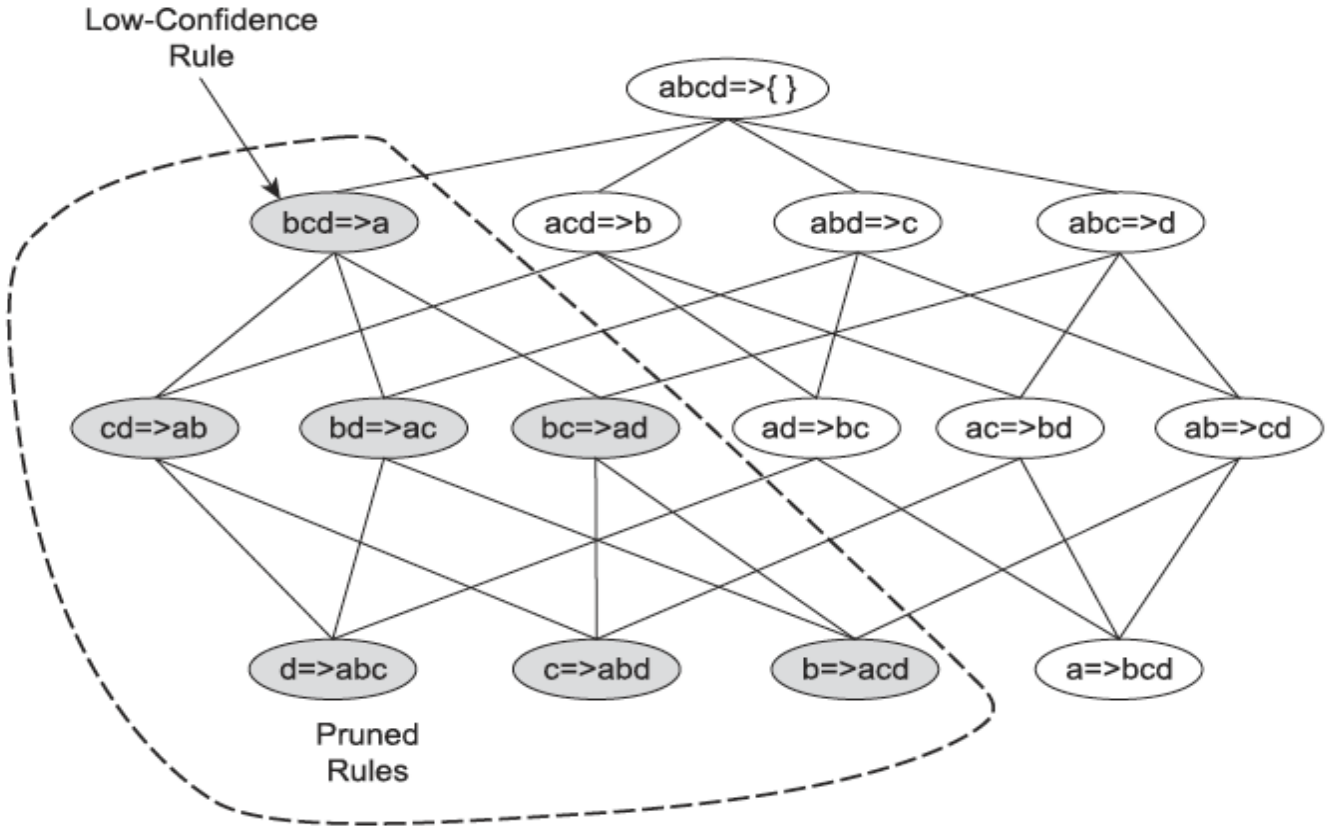


Figure 4.15. Pruning of association rules using the confidence measure.

Algorithm 4.2 Rule generation of the *Apriori* algorithm.

1: **for** each frequent k-itemset f_k, $k \geq 2$ **do**
2: $H_1 = \{i \mid i \in f_k\}$ {1-item consequents of the rule.}
3: call ap-genrules$(f_k, H_1.)$
4: **end for**

Algorithm 4.3 Procedure **ap-genrules**(f_k, H_m).

1: $k = |f_k|$ {size of frequent itemset.}
2: $m = |H_m|$ {size of rule consequent.}
3: **if** $k > m + 1$ **then**
4: H_{m+1} = candidate-gen(H_m).
5: H_{m+1} = candidate-prune(H_{m+1},H_m).
6: **for** each $h_{m+1} \in H_{m+1}$ **do**
7: $conf = \sigma(f_k)/\sigma(f_k - h_{m+1})$.
8: **if** $conf \geq minconf$ **then**
9: **output** the rule $(f_k - h_{m+1}) \longrightarrow h_{m+1}$.
10: **else**
11: **delete** h_{m+1} from H_{m+1}.
12: **end if**
13: **end for**
14: **call** ap-genrules(f_k, H_{m+1}.)
15: **end if**

•The key difference from frequent itemset generation is that confidence computation for candidate rules doesn't require additional passes over the dataset; instead, it utilizes support counts obtained during frequent itemset generation.

Application Example: Congressional Voting Records

Dataset Description

- Demonstrates applying association analysis to the voting records of U.S. House Representatives using the 1984 Congressional Voting Records Database from the UCI machine learning data repository.
- Each transaction holds party affiliation and voting records on 16 key issues.

Attributes in the Dataset

- A table (Table 4.3) listing the 34 binary attributes from the 1984 United States Congressional Voting Records dataset.

Extracted Association Rules

- After applying the Apriori algorithm with certain support and confidence thresholds, high-confidence rules are extracted and displayed in Table 4.4.
- These rules demonstrate the association between specific voting patterns and political party affiliations.

Table 4.3. List of binary attributes from the 1984 United States Congressional Voting Records. Source: The UCI machine learning repository.

1. Republican
2. Democrat
3. handicapped-infants = yes
4. handicapped-infants = no
5. water project cost sharing = yes
6. water project cost sharing = no
7. budget-resolution = yes
8. budget-resolution = no
9. physician fee freeze = yes
10. physician fee freeze = no
11. aid to El Salvador = yes
12. aid to El Salvador = no
13. religious groups in schools = yes
14. religious groups in schools = no
15. anti-satellite test ban = yes
16. anti-satellite test ban = no
17. aid to Nicaragua = yes
18. aid to Nicaragua = no
19. MX-missile = yes
20. MX-missile = no
21. immigration = yes
22. immigration = no
23. synfuel corporation cutback = yes
24. synfuel corporation cutback = no
25. education spending = yes
26. education spending = no
27. right-to-sue = yes
28. right-to-sue = no
29. crime = yes
30. crime = no
31. duty-free-exports = yes
32. duty-free-exports = no
33. export administration act = yes
34. export administration act = no

Table 4.4. Association rules extracted from the 1984 United States Congressional Voting Records.

Association Rule	Confidence
{budget resolution = no, MX-missile=no, aid to El Salvador = yes } $\longrightarrow$ {Republican}	91.0%
{budget resolution = yes, MX-missile=yes, aid to El Salvador = no } $\longrightarrow$ {Democrat}	97.5%
{crime = yes, right-to-sue = yes, physician fee freeze = yes} $\longrightarrow$ {Republican}	93.5%
{crime = no, right-to-sue = no, physician fee freeze = no} $\longrightarrow$ {Democrat}	100%

Compact Representation of Frequent Itemsets

In data mining and association rule analysis, discovering frequent itemsets from transactional data is a crucial step.

However, in real-world scenarios, this process often generates an extensive list of frequent itemsets, which might pose challenges in terms of managing, analyzing, and interpreting the results due to their sheer volume.

Hence, it becomes beneficial to find a more compact or concise representation of frequent itemsets that captures the essential information without overwhelming detail.

Two such representations, namely maximal and closed frequent itemsets

Maximal Frequent Itemsets:

- **Definition:** A frequent itemset is maximal if none of its immediate supersets are frequent. In simpler terms, it's a set that cannot be further extended to a larger frequent itemset.
- **Purpose:** Maximal frequent itemsets serve as a small representative subset that encapsulates all other frequent itemsets. They provide the smallest set of itemsets from which all frequent itemsets can be derived.
- **Utility:** They offer a more concise representation of frequent itemsets and help reduce the complexity associated with managing and analyzing extensive lists of frequent itemsets.
- **Application:** Maximal frequent itemsets are particularly useful when dealing with datasets that produce a large number of frequent itemsets, preventing the need to explicitly list all frequent itemsets by representing only those that cannot be further expanded.

Closed Frequent Itemsets:

- **Definition:** A closed itemset is closed if none of its immediate supersets has the same support count as itself. It represents a minimal representation of itemsets that retains support information.
- **Purpose:** Closed frequent itemsets aim to offer a minimal yet informative representation of itemsets by preserving their support information. They capture the essential information without redundancy.
- **Utility:** Closed frequent itemsets are valuable when there's a need for a compact representation that maintains the support information of itemsets. They efficiently convey relevant information about frequent itemsets while minimizing redundancy.
- **Application:** They are particularly handy when dealing with datasets where maintaining support information for itemsets is crucial for analysis or decision-making processes.

Maximal Frequent Itemsets:

- **Definition**:
  - Maximal Frequent Itemset: An itemset is maximal if none of its immediate supersets are frequent.
  - Consider an itemset lattice divided into frequent and infrequent sets with a border delineating frequent itemsets.
  - Maximal frequent itemsets are those whose immediate supersets are all infrequent.
  - Maximal frequent itemsets provide a compact representation from which all frequent itemsets can be derived.
- **Benefits**:
  - They represent the smallest set of itemsets from which all frequent itemsets can be derived.
  - Enumerating subsets of maximal frequent itemsets generates the complete list of all frequent itemsets.
- **Limitation**:
  - To determine the support counts of non-maximal frequent itemsets, an additional pass over the dataset is required.

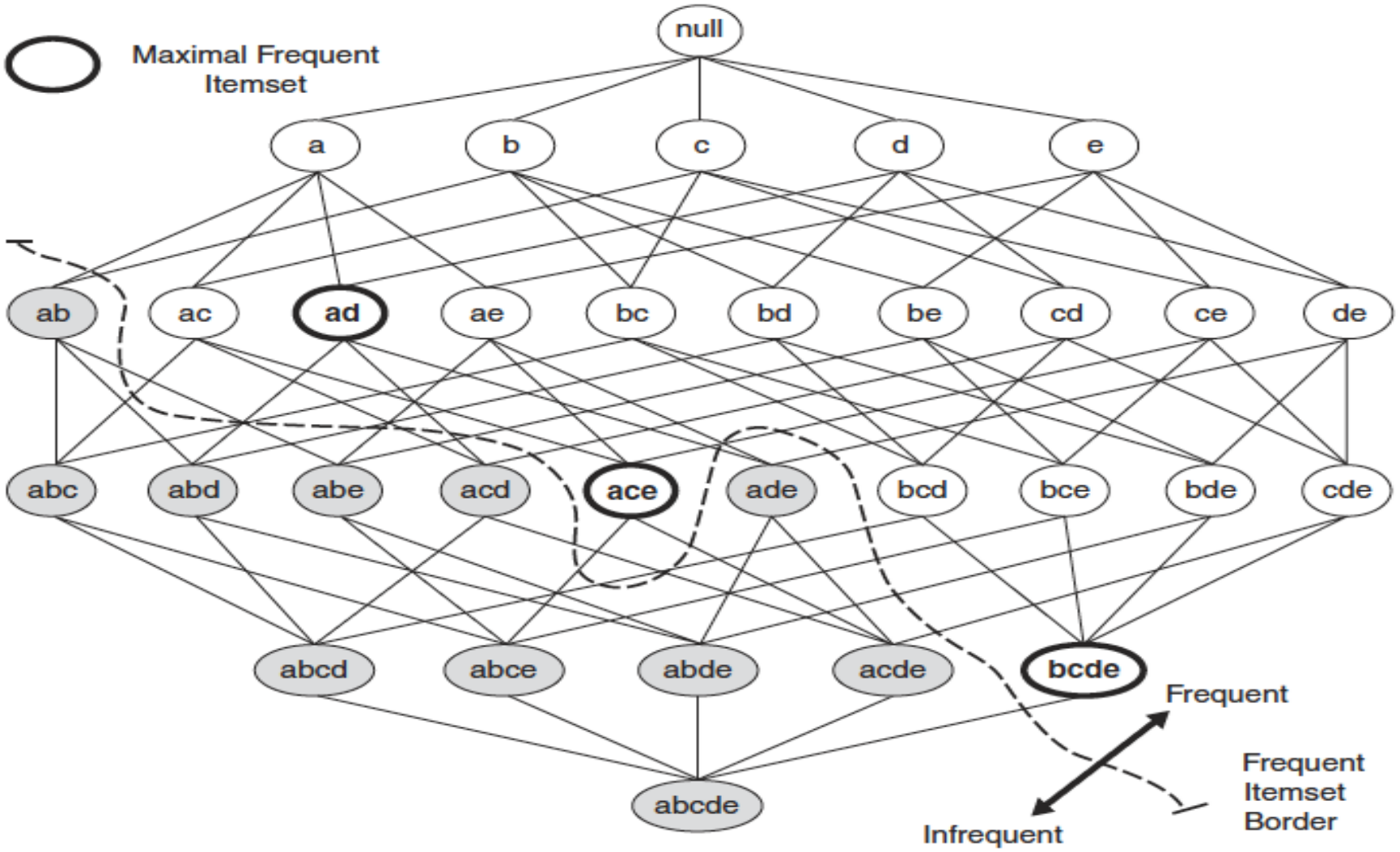


Figure 4.16. Maximal frequent itemset.

Closed Itemsets

- **Definition**:
  - Closed Itemset: An itemset X is closed if none of its immediate supersets has the same support count as X.
  - Examples provided in a lattice structure showcasing closed and non-closed itemsets based on their support counts.
  - Closed itemsets provide a minimal representation without losing support information.
  - Knowing the support counts of closed itemsets can help derive the support counts of other itemsets without additional passes over the dataset.
- **Advantages**:
  - Closed itemsets preserve support information for all itemsets.
  - They offer a way to compute support counts without additional scans of the dataset.
- **Application**:
  - Closed frequent itemsets, being both closed and frequent, provide a compact representation of support counts for all frequent itemsets.
  - Maximal frequent itemsets are also closed since none of them can have the same support count as their immediate supersets.

Algorithm 4.4 Support counting using closed frequent itemsets.

1: Let C denote the set of closed frequent itemsets and F denote the set of all frequent itemsets.
2: Let $k_{\max}$ denote the maximum size of closed frequent itemsets
3: $F_{k_{\max}} = \{f | f \in C,\ |f| = k_{\max}\}$ {Find all frequent itemsets of size $k_{\max}$.}
4: **for** $k = k_{\max} - 1$ down to 1 **do**
5: $F_k = \{f | f \in F,\ |f| = k\}$ {Find all frequent itemsets of size k.}
6: **for** each $f \in F_k$ **do**
7: **if** $f \notin C$ **then**
8: $f.support = \max\{f'.support | f' \in F_{k+1},\ f \subset f'\}$
9: **end if**
10: **end for**
11: **end for**

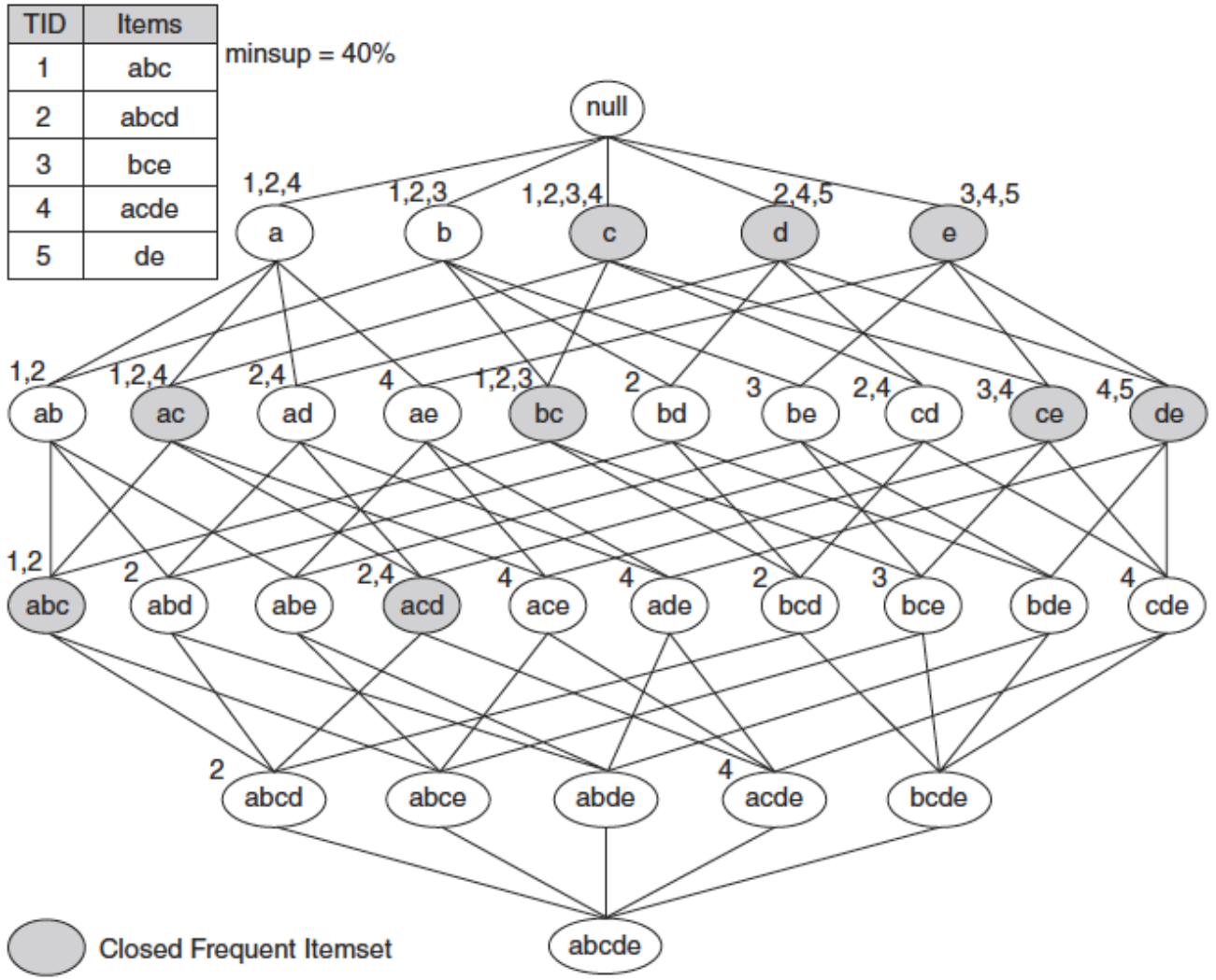


Figure 4.17. An example of the closed frequent itemsets (with minimum support equal to 40%).

ADVANTAGES

- To illustrate the advantage of using closed frequent itemsets, consider the data set shown in Table 4.5, Contains ten transactions and fifteen items.
- The items can be divided into three groups:
  1. Group A, which contains items a1 through a5;
  2. Group B, which contains items b1 through b5;
  3. Group C, which contains items c1 through c5.
- Assuming that the support threshold is 20%, itemsets involving items from
- the same group are frequent, but itemsets involving items from different groups are infrequent. The total number of frequent itemsets is thus 3 X (2^5 – 1) = 93.
- However, there are only four closed frequent itemsets in the data:
  - ({a3, a4}, {a1, a2, a3, a4, a5}, {b1, b2, b3, b4, b5}, and {c1, c2, c3, c4, c5}).
- It is often sufficient to present only the closed frequent itemsets to the analysts instead of the entire set of frequent itemsets.

Table 4.5. A transaction data set for mining closed itemsets.

TID	a_1	a_2	a_3	a_4	a_5	b_1	b_2	b_3	b_4	b_5	c_1	c_2	c_3	c_4	c_5
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	1	1	0	1	1	1	1	1	0	0	0	0	0
5	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
6	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
8	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
9	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
10	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

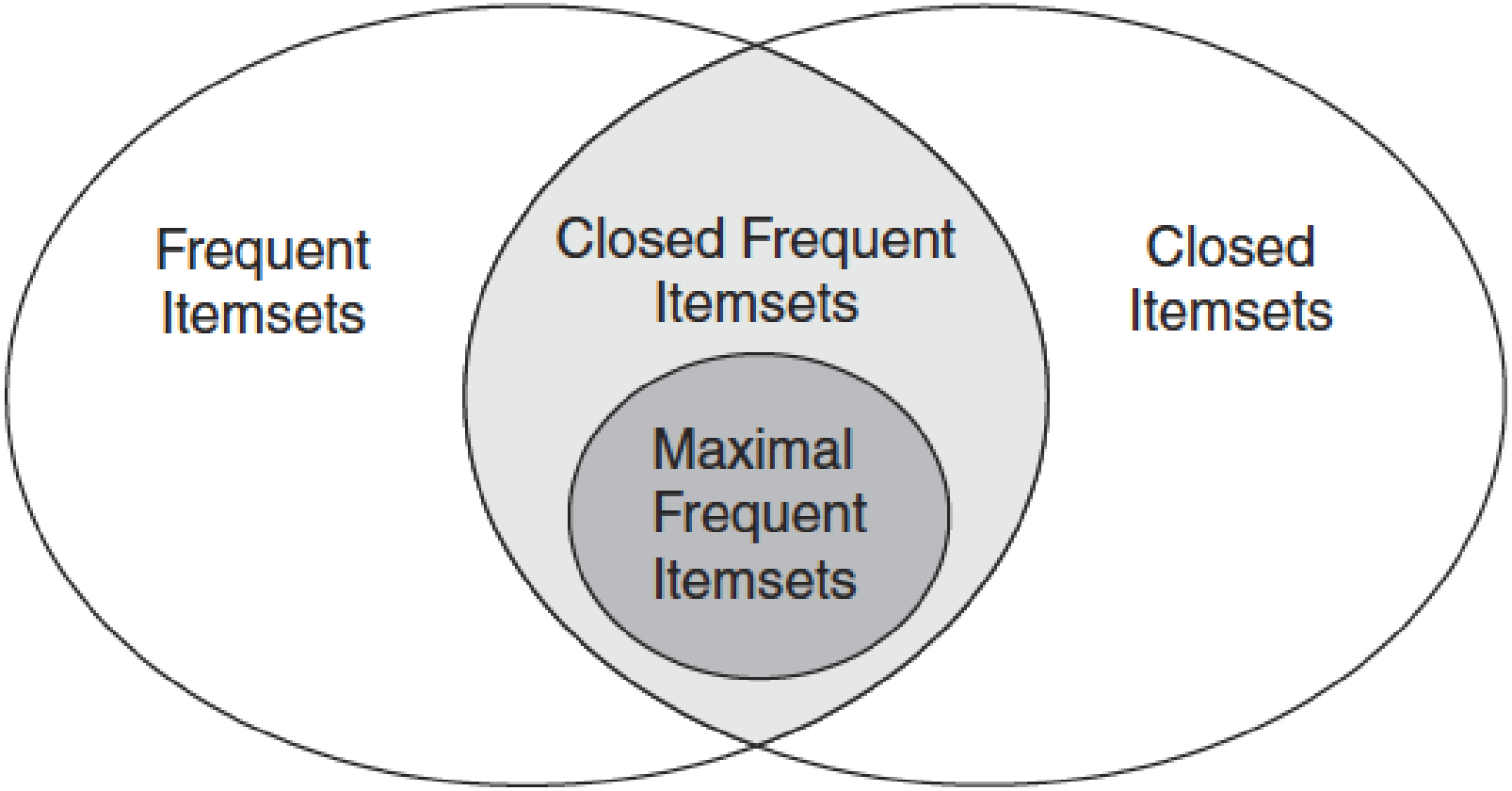


Figure 4.18. Relationships among frequent, closed, closed frequent, and maximal frequent itemsets.