Mining Associations

Apriori Algorithm

Adapted from two sources of slides: J. D. Ullman, and V. Kumar

Co-occurrence mining

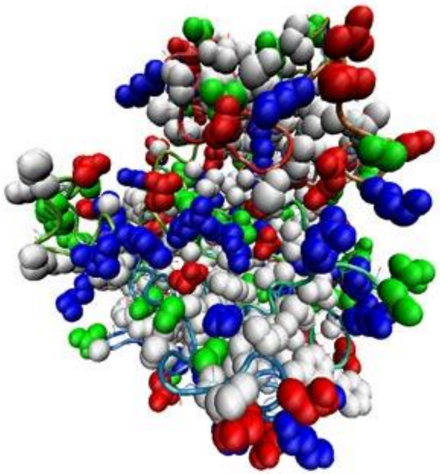
Conceptually simple
practically hard!

Learn sets of items that frequently show up together.

Billions of documents
Hundreds of thousands of words

Surprising associations

Scale of Problem



Example: Frequent Itemsets

- Items={**m**ilk, **c**oke, **p**epsi, **b**eer, **j**uice}.
- Support count threshold = 3 baskets.

  B_1 = {m, c, b} B_2 = {m, p, j}
  B_3 = {m, b} B_4 = {c, j}
  B_5 = {m, p, b} B_6 = {m, c, b, j}
  B_7 = {c, b, j} B_8 = {b, c}

- Frequent itemsets: {m}, {c}, {b}, {j},

  {m,b}, {b,c} , {c,j}.

Association Rules

- If-then rules about the contents of baskets.
- $\{i_1, i_2,\ldots,i_k\} \rightarrow j$ means: "if a basket contains all of $i_1,\ldots,i_k$ then it is *likely* to contain j."
- *Confidence* of this association rule is the probability of j given $i_1,\ldots,i_k$.
  - confidence= support($i_1,\ldots,i_k$ j) / support($i_1,\ldots,i_k$)

Example

B_1 = {m, c, b} B_2 = {m, p, j}

B_3 = {m, b} B_4 = {c, j}

B_5 = {m, p, b} B_6 = {m, c, b, j}

B_7 = {c, b, j} B_8 = {b, c}

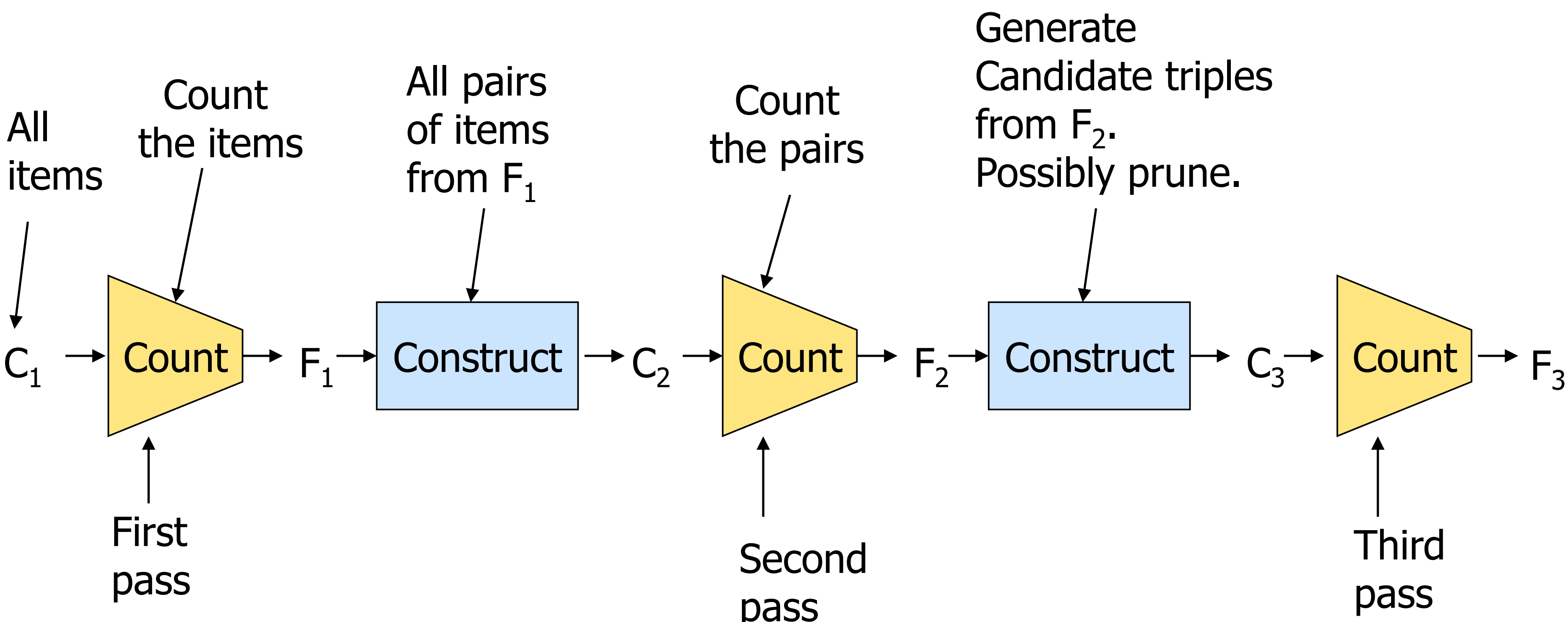
- An association rule: {m, b} → c.
  - Confidence = 2/4 = 50%.

Apriori Principle

- If an itemset A is frequent, then each itemset $B \subset A$ is frequent.
- Given an itemset A, if we can find an itemset $B \subset A$ that's not frequent, then A cannot be frequent.

Apriori Algorithm

- For each k, we construct two sets of k–itemsets:
  - C_k = candidate k - itemsets = those that might be frequent (support $\geq s$) based on information from the pass for $k-1$.
  - F_k = the set of truly frequent k - itemsets.



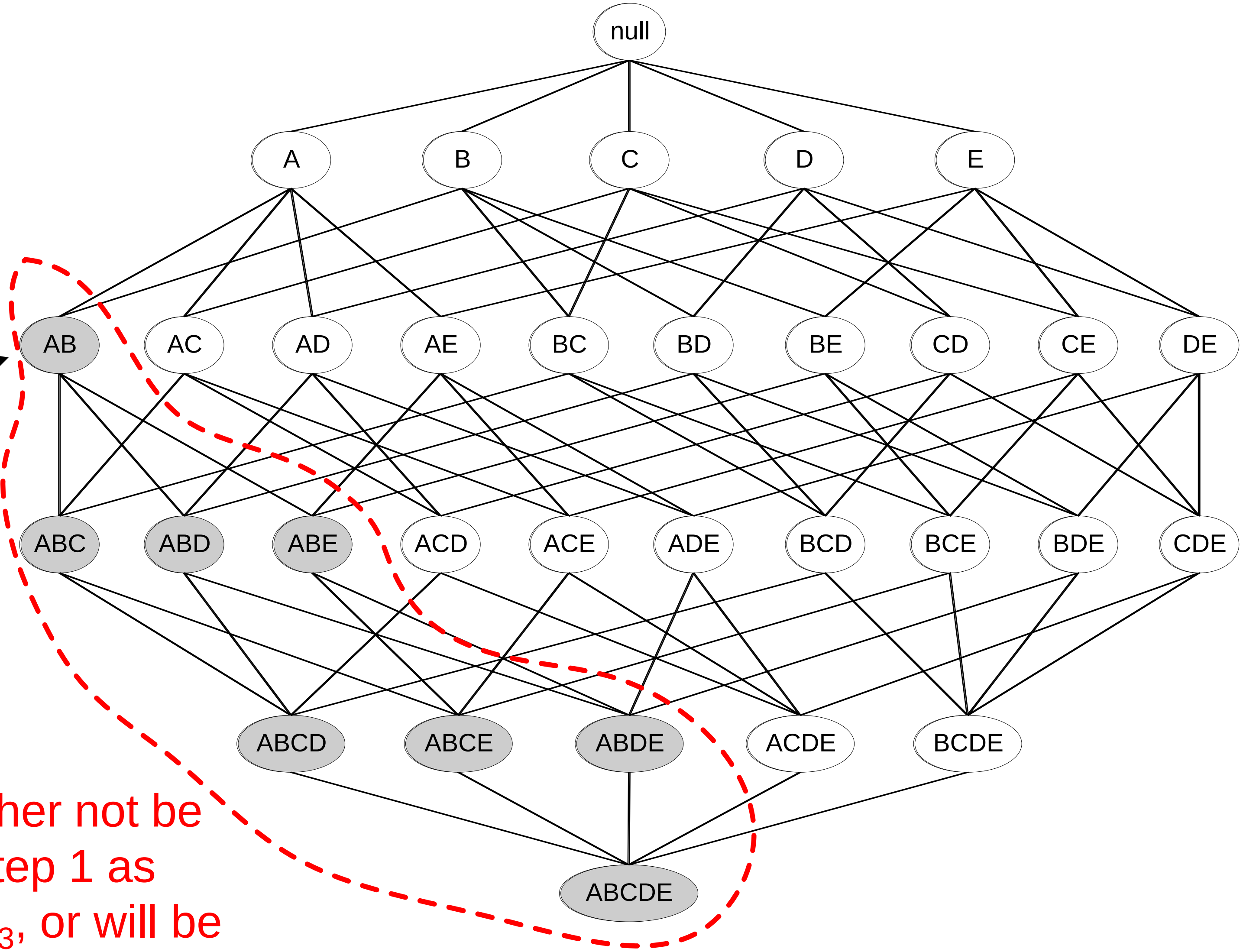
Apriori Algorithm

- Let k=1
- Generate frequent itemsets of length 1
- Repeat until no new frequent itemsets are found

  k=k+1

  1. **Generate** length k candidate itemsets from length k-1 frequent itemsets
  2. **Prune** candidate itemsets containing subsets of length k-1 that are infrequent
  3. **Count** the support of each candidate by scanning the DB and eliminate candidates that are infrequent, leaving only those that are frequent

Benefit of the Apriori principle



Suppose AB is not in F_2.

All these will either not be generated by Step 1 as candidates in C_3, or will be pruned in Step 2.

Data Set Example

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**

s=3

Candidate generation: $F_{k-1} \times F_1$ Method

- Extend each frequent (*k*-1)-itemset with a frequent 1-itemset.
- However, it doesn't prevent the same candidate itemset from being generated more than once.
  - E.g., {Bread, Diapers, Milk} can be generated by merging
  - {Bread, Diapers} with {Milk},
  - {Bread, Milk} with {Diapers}, or
  - {Diapers, Milk} with {Bread}.

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 1-itemset

Item
Beer
Bread
Diapers
Milk

Lexicographic Order

- Keep frequent itemset sorted in lexicographic order.
- Each frequent (k-1)-itemset X is extended with frequent items that are lexicographically larger than the items in X.

Example

- {Bread, Diapers} can be extended with {Milk}
- {Bread, Milk} can't be extended with {Diapers}
- {Diapers, Milk} can't be extended with {Bread}

Pruning

- Merging {Beer, Diapers} with {Milk} is unnecessary. Why?
- Because one of its subsets, {Beer, Milk}, is infrequent.
- Solution: Prune!
- How?
- Check each k-1 subset of the candidate created.
- If one of them is infrequent, prune candidate.

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 1-itemset

Item
Beer
Bread
Diapers
Milk

$F_{k-1} \times F_{k-1}$ Method

- Merge a pair of frequent (k-1) itemsets only if their first k-2 items are identical.

- E.g. frequent itemsets

  {Bread, Diapers} and {Bread, Milk}

  are merged to form a candidate 3 itemset

  {Bread, Diapers, Milk}.

$F_{k-1} \times F_{k-1}$ Method

- We don't merge {Beer, Diapers} with {Diapers, Milk} because the first item in both itemsets is different.

But, is this "don't merge" decision Ok?

- Indeed, if {Beer, Diapers, Milk} is a viable candidate, it would have been obtained by merging {Beer, Diapers} with {Beer, Milk} instead.

Pruning

- Because each candidate is obtained by merging a pair of frequent (k-1)itemsets, an additional candidate pruning step is needed to ensure that the remaining k-2 subsets of k-1 elements are frequent.

$F_{k-1} \times F_{k-1}$ Example

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Frequent 2-itemset

Itemset
{Beer, Diapers}
{Bread, Diapers}
{Bread, Milk}
{Diapers, Milk}

Candidate Generation

Itemset
{Bread, Diapers, Milk}

Another Example

Min_sup_count = 2

TID	List of item ID's
T1	1, 2, 5
T2	2, 4
T3	2, 3
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

C1

Itemset
{1}
{2}
{3}
{4}
{5}

F1

Itemset	Sup. count
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2

Generate C2 from F1×F1

Min_sup_count = 2

TID	List of item D’s
T1	1, 2, 5
T2	2, 4
T3	2, 3
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

F1

Itemset	Sup. count
{1}	6
{2}	7
{3}	6
{4}	2
{5}	2

C2

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

Itemset	Sup. C
{1,2}	4
{1,3}	4
{1,4}	1
{1,5}	2
{2,3}	4
{2,4}	2
{2,5}	2
{3,4}	0
{3,5}	1
{4,5}	0

Generate C3 from F2×F2

Min_sup_count = 2

TID	List of item ID's
T1	1, 2, 5
T2	2, 4
T3	2, 3
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

F2

Itemset	Sup. C
{1,2}	4
{1,3}	4
{1,5}	2
{2,3}	4
{2,4}	2
{2,5}	2

C3

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

Prune

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

F3

Itemset	Sup. C
{1,2,3}	2
{1,2,5}	2

Generate C4 from F3×F3

Min_sup_count = 2

TID	List of item ID’s
T1	1, 2, 5
T2	2, 4
T3	2, 3
T4	1, 2, 4
T5	1, 3
T6	2, 3
T7	1, 3
T8	1, 2, 3, 5
T9	1, 2, 3

C4

Itemset
{1,2,3,5}

{1,2,3,5} is pruned because {2,3,5} is infrequent

F3

Itemset	Sup. C
{1,2,3}	2
{1,2,5}	2

Compact Representation

Need for Compact Representation of Frequent Itemsets

TID	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

- Number of frequent itemsets (supp. count 5)
- Need a compact representation

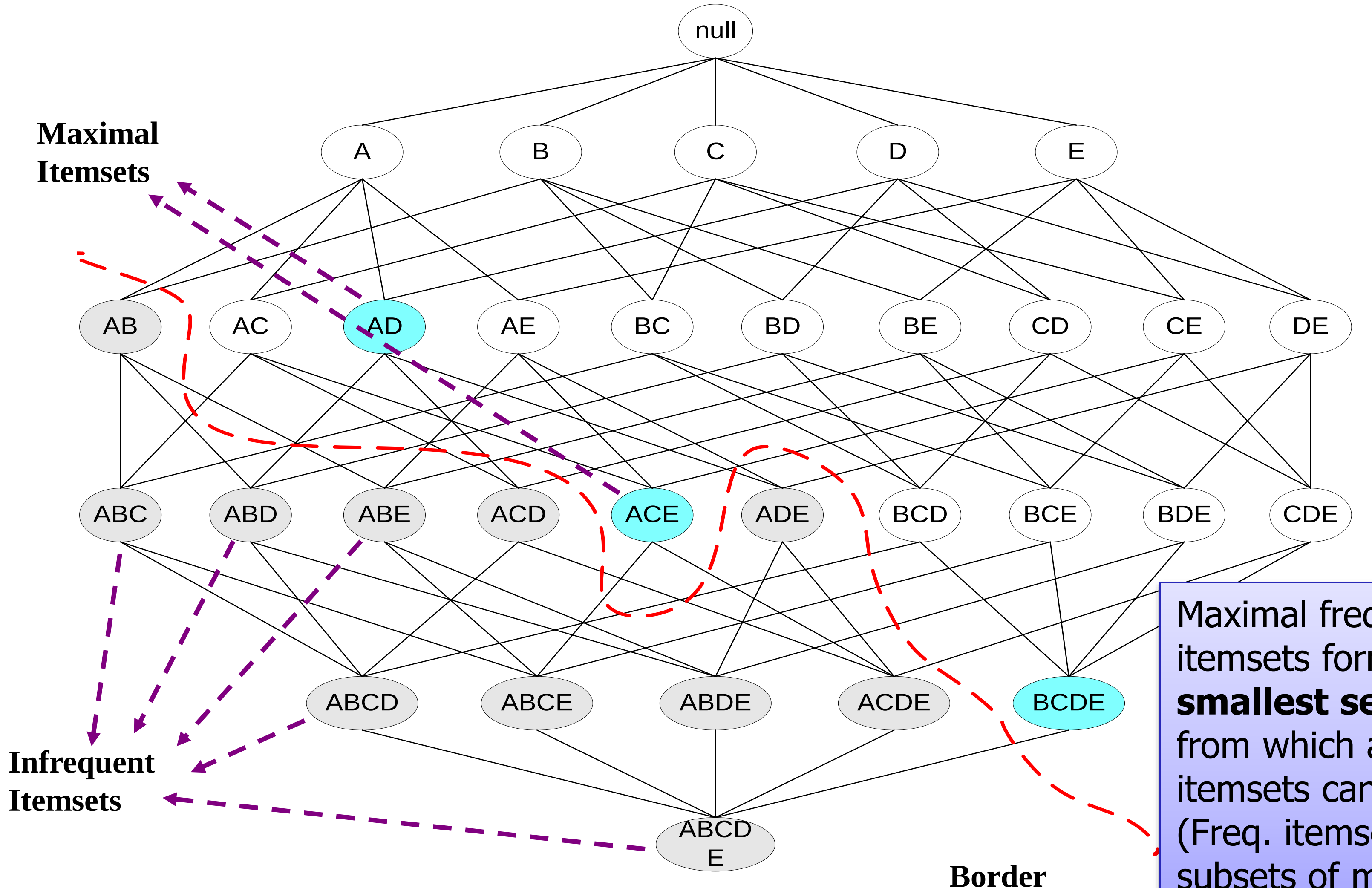
$$= 3 \times \sum_{k=1}^{10} \binom{10}{k}$$

$$= 3 \times \left(2^{10} - 1\right)$$

$$= 3069$$

Maximal Frequent Itemsets

A freq. itemset is **maximal freq.** if none of its immediate supersets is frequent



Maximal frequent itemsets form the **smallest set** of itemsets from which all frequent itemsets can be derived. (Freq. itemsets are the subsets of maximal itemsets)

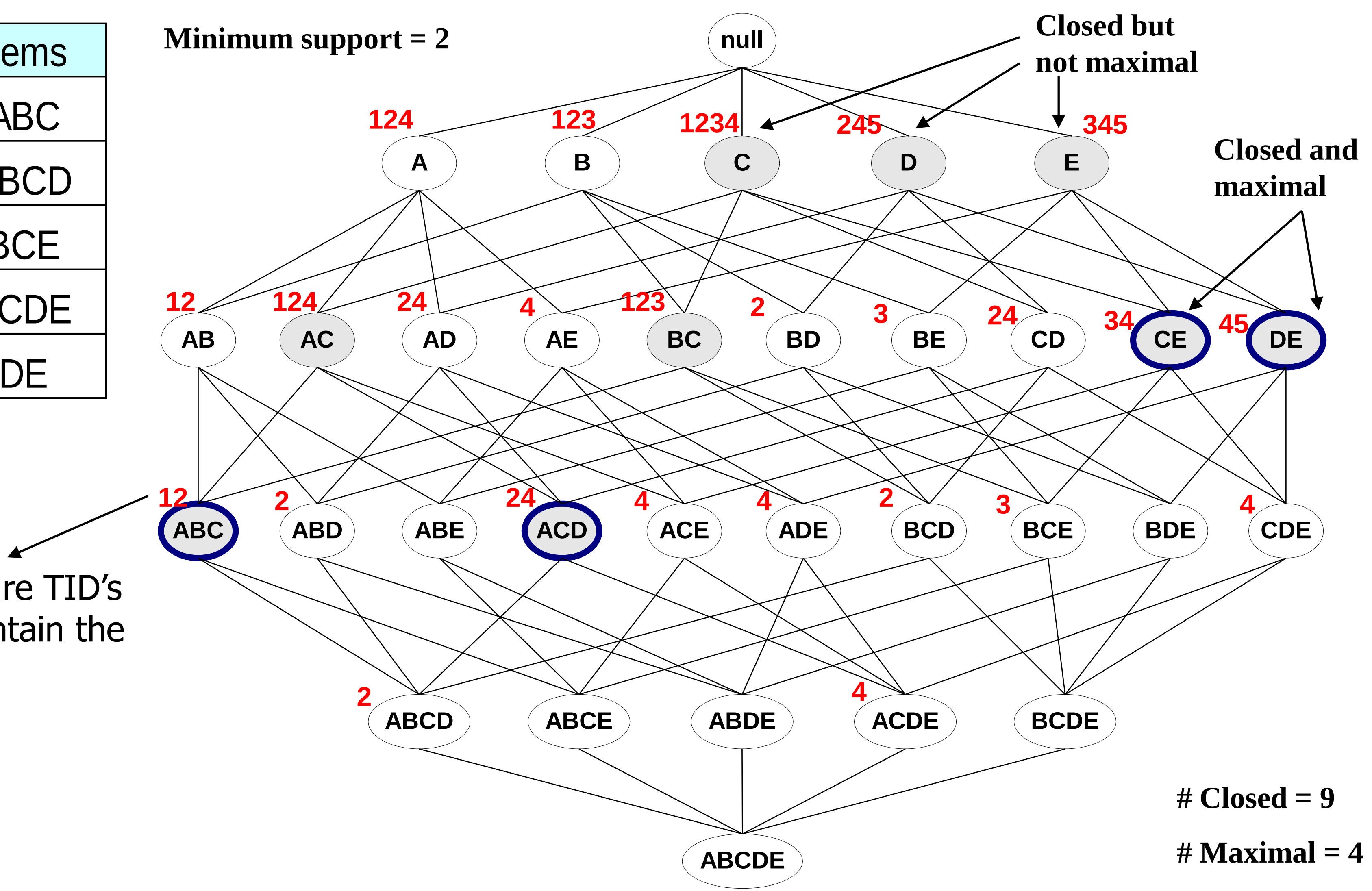
Maximal Frequent Itemsets - Problem

While we can derive all frequent itemsets from maximal ones, we can’t determine the support count of the frequent itemsets.

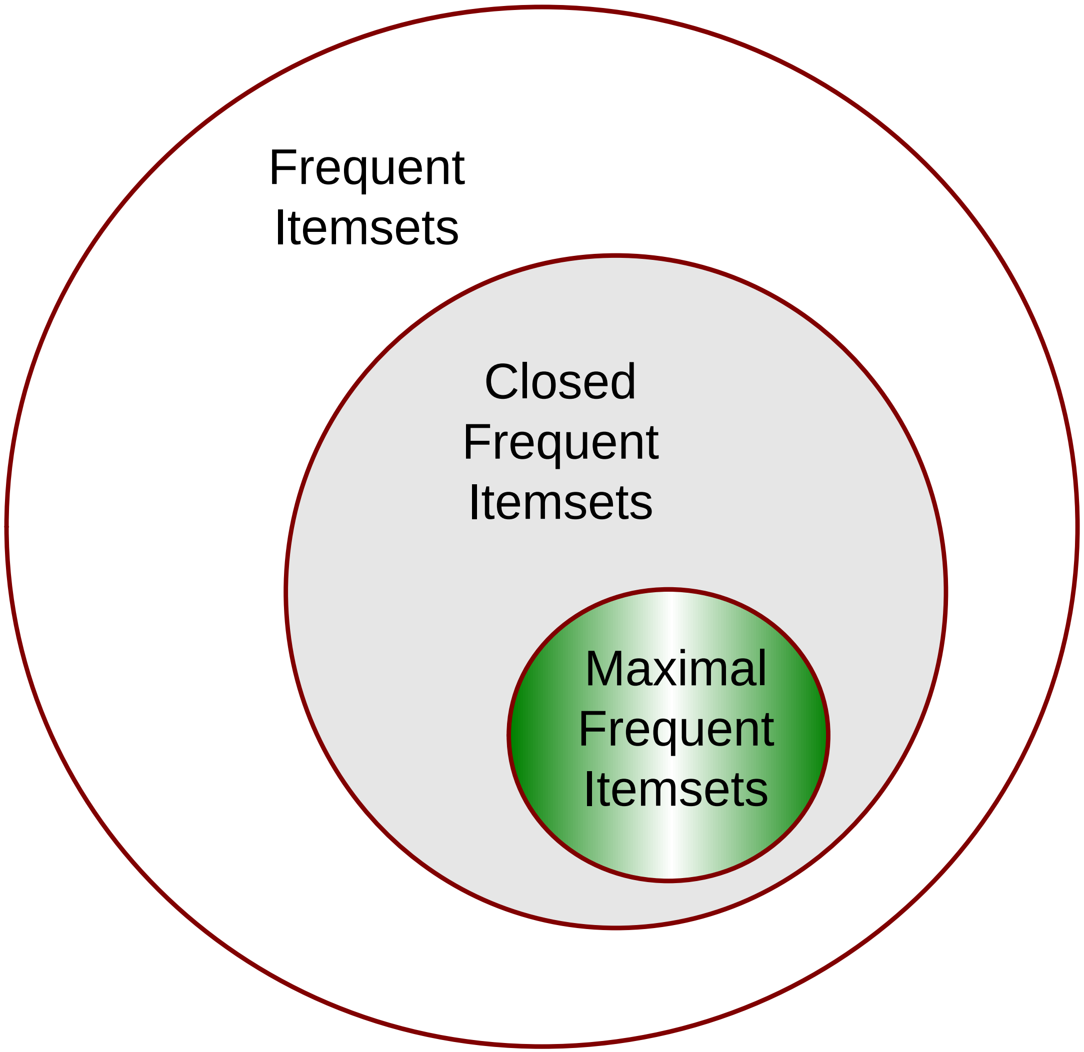
Closed Frequent Itemsets

A freq. itemset is **closed frequent** if none of its immediate supersets has same supp.

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



Maximal vs Closed Itemsets



Deriving Frequent Itemsets From Closed Frequent Itemsets

- Consider a **frequent** itemset *itset* that is **not closed**,

  i.e. there exists a **superset** of *itset* that is **frequent** and **closed** and has **the same support as** *itset*.

  maybe **more than one such superset**.

- **Question**: Which one of supersets of *itset* has the same support as *itset* ?

- **Answer**: The support of *itset* must be equal to the **largest support** among its closed supersets.
  - Why? Because of the apriori principle. The subset should have at least the support of the superset.

Example

Closed = {ABC:3, ACD:4, CE:6, DE:7}

F3 = {ABC:3, ACD:4}

F2 = {AB:3, AC:4, BC:3, AD:4, CD:4, CE:6, DE:7}

F1 = {A:4, B:3, C:6, D:7, E:7}

Computing Frequent Closed Itemsets

- Use the Apriori Algorithm.
- After computing, say F_k and F_{k+1},
  - Check for itemsets in F_k that have a support equal to the support of one of their supersets in F_{k+1}.
  - Purge all such itemsets from F_k.